

Gesellschaft für Informatik e.V. (GI)

publishes this series in order to make available to a broad public recent findings in informatics (i.e. computer science and information systems), to document conferences that are organized in cooperation with GI and to publish the annual GI Award dissertation.

Broken down into

- seminars
- proceedings
- dissertations
- thematics

current topics are dealt with from the vantage point of research and development, teaching and further training in theory and practice. The Editorial Committee uses an intensive review process in order to ensure high quality contributions.

The volumes are published in German or English.

Information: <http://www.gi.de/service/publikationen/lni/>

ISSN 1617-5468

ISBN 978-3-88579-283-3

“INFOS 2011” is the 14th event in a conference series organized by the GI special interest group IBS focusing on general education in informatics (computer science) in schools.

The present volume contains contributions accepted by the program committee and short papers of invited talks.



M. Thomas (Hrsg.): Informatik in Bildung und Beruf – INFOS 2011

GI-Edition

Lecture Notes in Informatics

Thomas, Marco (Hrsg.)

Informatik in Bildung und Beruf

INFOS 2011

14. GI-Fachtagung Informatik und Schule

12.–15. September 2011 in Münster

Proceedings





Marco Thomas (Hrsg.)

Informatik in Bildung und Beruf

**14. GI-Fachtagung „Informatik und Schule – INFOS 2011“
12.-15. September 2011 an der
Westfälischen Wilhelms-Universität Münster**

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Proceedings

Series of the Gesellschaft für Informatik (GI)

Volume P-189

ISBN 978-3-88579-283-3

ISSN 1617-5468

Volume Editors

Prof. Dr. Marco Thomas

Westfälische Wilhelms-Universität Münster

Fachbereich Mathematik und Informatik

Institut für Didaktik der Mathematik und der Informatik

Fliednerstraße 21

48149 Münster

E-Mail: thomasma@uni-muenster.de

Series Editorial Board

Heinrich C. Mayr, Universität Klagenfurt, Austria (Chairman, mayr@ifit.uni-klu.ac.at)

Hinrich Bonin, Leuphana-Universität Lüneburg, Germany

Dieter Fellner, Technische Universität Darmstadt, Germany

Ulrich Flegel, Hochschule Offenburg, Germany

Ulrich Frank, Universität Duisburg-Essen, Germany

Johann-Christoph Freytag, Humboldt-Universität Berlin, Germany

Thomas Roth-Berghofer, DFKI, Germany

Michael Goedicke, Universität Duisburg-Essen, Germany

Ralf Hofestädt, Universität Bielefeld, Germany

Michael Koch, Universität der Bundeswehr, München, Germany

Axel Lehmann, Universität der Bundeswehr München, Germany

Ernst W. Mayr, Technische Universität München, Germany

Sigrid Schubert, Universität Siegen, Germany

Martin Warnke, Leuphana-Universität Lüneburg, Germany

Dissertations

Steffen Hölldobler, Technische Universität Dresden, Germany

Seminars

Reinhard Wilhelm, Universität des Saarlandes, Germany

Thematics

Andreas Oberweis, Universität Karlsruhe (KIT), Germany

© Gesellschaft für Informatik, Bonn 2011

printed by Köllen Druck+Verlag GmbH, Bonn

Vorwort

Die Informatik ist mit ihren Produkten unbestreitbar einer der wichtigsten “Motoren” in unserer Gesellschaft. Manche sprechen sogar von einer »Digitalen Revolution«, die nach den Phasen der Industrialisierung einsetzte. Die industrielle Revolution hat zur Etablierung der naturwissenschaftlichen Fächer im Schulkanon geführt, die Digitalisierung zum Informatikunterricht.

Die föderale Struktur des deutschen Bildungssystems ist eine Ursache der recht mannigfaltigen Ausgestaltung des Schulfachs Informatik. Während beispielsweise in der gymnasialen Oberstufe – auf Basis der einheitlichen Prüfungsanforderungen der Kultusminister für das Abitur – kerninformatische Inhalte bundesweit dominieren, zeigt der Informatikunterricht in den Mittelstufen der Länder ein breites inhaltliches Spektrum, das oft von einer medienorientierten Anwendung von Software bestimmt wird.

Moderne Medien basieren auf Informatiksystemen, deren Beherrschung in vielen Berufen erforderlich ist; möglicherweise vergleichbar mit grundlegenden Fähigkeiten in den Kulturtechniken des Lesens, Schreibens und Rechnens. Dass proprietäre, externe Zertifikate, die zunehmend eingesetzt werden, um Fertigkeiten zur Bedienung von Informatiksystemen festzustellen, für eine allgemein bildende Schule geeignet sind, darf allerdings stark bezweifelt werden. Zumindest ist mir kein anderes Schulfach mit allgemein bildenden Anspruch bekannt, das vergleichbare Zertifikate einsetzt.

Aktuell wird in mehreren Bundesländern eine Kombination von Informatik und Medienkunde diskutiert. Die bildungspolitischen Entscheidungsträger würden die Einrichtung eines fächerverbindenden Faches »Informatik und Medien« o. ä. voraussichtlich auch stärker unterstützen als ein an der Wissenschaft Informatik orientiertes Schulfach. Zu diskutieren ist jedoch, inwieweit Informatische Bildung – deren Notwendigkeit mehr und mehr durch das aktuelle Tagesgeschehen deutlich wird – in einem solchen Fach erworben werden kann. Ein ähnlicher, fächerintegrierter Ansatz einer informationstechnischen Grundbildung (ITG u. a.) ist im letzten Jahrhundert bereits gescheitert. Dass Informatikunterricht einen Beitrag zur Medienbildung leisten kann und will, konnte bereits den im Jahr 2000 erschienenen Empfehlungen »Informatische Bildung und Medienerziehung« der Gesellschaft für Informatik e.V (GI) entnommen werden. Die INFOS wird sich intensiv an diesen Diskussionen beteiligen.

Einen bundesweiten Konsens zu Mindestanforderungen an Informatikunterricht als Grundlage einer Informatischen Bildung stellen die 2008 publizierten »Empfehlungen der Gesellschaft für Informatik e.V. (GI) zu Bildungsstandards Informatik für die Sekundarstufe I« dar. Deren Weiterentwicklung, unterrichtspraktische Ausgestaltung und die begleitenden Forschungsansätze sind weitere Schwerpunkte der Diskussionen auf der INFOS. Mittlerweile beginnen sich die Standards auch auf die gymnasiale Oberstufe auszuwirken, so dass möglicherweise die Breite des Fachs Informatik in den höheren Jahrgangstufen besser abgebildet werden kann. Ein Zentralabitur darf diesem Gedanken und der Weiterentwicklung des Schulfachs Informatik nicht entgegenwirken.

Informatiksysteme und ihre Phänomene sind ebenso allgegenwärtig wie die Phänomene, die in den Naturwissenschaften betrachtet werden. Insofern erscheint es konsequent, wenn eine »Informatik im Kontext« (IniK) für den Informatikunterricht entwickelt wird, vergleichbar mit den Bestrebungen in den Naturwissenschaften. In diesem Konzept dürfte auch die lange geforderte Integration von schulrelevanten Themen aus dem Teilgebiet »Informatik und Gesellschaft« möglich sein. Ähnlich wie bei der Entwicklung der Bildungsstandards gibt es bei IniK auch erste Ansätze für die Oberstufeninformatik.

Die oben skizzierten Entwicklungen zum Informatikunterricht zeigen, dass eine Oberstufeninformatik natürlicherweise auf einem Pflichtfach Informatik in der Sekundarstufe I aufbauen sollte. Zur schulischen Allgemeinbildung gehört eine an den naturwissenschaftlichen Fachdisziplinen orientierte Sichtweise ebenso wie informatisches Denken und Arbeiten. Das Programmieren ist – im Sinne von ein informatisches Modell entwerfen und implementieren – als fachwissenschaftliche Methode für den Unterricht von ähnlicher Bedeutung wie die naturwissenschaftliche Methode des Experimentierens. Die Vor- und Nachteile bestimmter Programmiersprachen und Lernumgebungen sind traditionell ein Thema auf der INFOS gewesen. In den letzten Jahren entstanden insbesondere für den Unterricht in der Mittelstufe lernförderliche Werkzeuge. Für das soziale Lernen in der Schule bietet das methodische Vorgehen in der Fachwissenschaft Informatik zahlreiche Anknüpfungspunkte (bspw. aus der agilen Softwareentwicklung), die teamorientierte und individuelle Kompetenzen ausbilden können. Beiträge aus der Fachwissenschaft können Impulse zur Lösung fachdidaktischer Probleme geben (bspw. zur Vereinbarkeit von pädagogischem und informatischem Projektbegriff).

Leider fehlen dem Schulfach Informatik seit Jahren grundständig ausgebildete Lehrer (und insbesondere Lehrerinnen). Auf der INFOS lassen sich Ursachen und Lösungen (bspw. zu einer stärker praxisorientierten Hochschullehre) erörtern. Fortbildungsangebote für Informatiklehrer – insbesondere in der Sekundarstufe I – sind rar. Workshops der INFOS und des integrierten NRW-Informatiktages geben Impulse für die Unterrichtspraxis.

Die Tagung »Informatik und Schule 2011« bietet Lehrkräften und Wissenschaftlern einen Rahmen zum Austausch von Ideen, Konzepten und Forschungsergebnissen, die in diesem Tagungsband und einem Praxisband gesammelt wurden. Das Motto »Informatik für Bildung und Beruf« soll zum einen fachdidaktische Überlegungen für allgemein bildende und stärker berufsspezifische Schulen zusammenbringen. Zum anderen soll verdeutlicht werden, dass das Schulfach Informatik sowohl einen allgemein bildenden als auch einen berufsvorbereitenden Beitrag leistet.

Bedanken möchte ich mich bei allen, die die INFOS ermöglicht haben, durch Präsentationen, Workshops, organisatorische Unterstützung und Sponsoring.

Ich wünsche uns eine erfolgreiche und einprägsame Tagung in Münster.

Münster, im September 2011

Marco Thomas

Mit freundlicher Unterstützung durch

LANCOM
Systems

und

 **MÜNSTER**
MARKETING



INFOS 2011

14. GI-Fachtagung »Informatik und Schule«

Informatik für Bildung und Beruf

Veranstaltendes Fachgremium der Gesellschaft für Informatik

Fachausschuss „Informatische Bildung in Schulen“

Programmkomitee

Marco Thomas (Universität Münster)

Norbert Breier (Universität Hamburg)
Torsten Brinda (Universität Erlangen)
Katrín Büttner (Mittelschule Heidenau)
Dieter Engbring (Universität Paderborn)
Steffen Friedrich (TU Dresden)
Michael Fothe (Universität Jena)
Werner Hartmann (PH Bern)
Ludger Humbert (Universität Wuppertal)
Peter Micheuz (Universität Klagenfurt)
Reinhard Oldenburg (Universität Frankfurt)
Johann Penon (OSZH I Berlin)
Hermann Puhlmann (Leibniz-Gymnasium Altdorf)
Ralf Romeike (Universität Potsdam)
Kirsten Schlüter (Universität Erlangen)
Carsten Schulte (FU Berlin)
Andreas Schwill (Universität Potsdam)
Franz Stuber (FH Münster)
Michael Weigend (Holzkamp-Gesamtschule Witten)

Organisation

Lilli Machleit
Frank Otte
Prof. Dr. Marco Thomas (Vorsitz)
Dr. Michael Weigend

Didaktik der Informatik
Westfälische Wilhelms-Universität
Münster
Fließerstraße 21
48149 Münster
<http://ddi.uni-muenster.de>
ddi@uni-muenster.de

Inhaltsverzeichnis

Eingeladene Vorträge

Stefan Jähnichen

Zehn Gründe Informatik zu studieren – Voraussetzungen,
Motivation und Vorbereitung in der Schule 13

Wolfgang Pohl

Informatik – kein Interesse? 15

Andreas Schreiber

Informatik für die Welt von Morgen 21

Stefanie Scherzinger

Agil und spielerisch: Neue Methoden der Software-
Entwicklung in der Praxis und ihr Potential für den Schulunterricht 23

Jochen Koubek

Die Medien der Informatik 25

Tagungsvorträge

Eckart Modrow

Visuelle Programmierung - oder: Was lernt man aus Syntaxfehlern? 27

Timo Göttel

Agiler Informatikunterricht: Soziale Aspekte der professionellen
Softwareentwicklung einfach und erfolgreich im Unterricht erfahrbar machen 37

Carsten Schulte

Das Schülerlabor als Ort der Informatiklehrerbildung 47

Bernd Bethge, Dirk Drews, Ursula Rump, Michael Fothe, Gabor Meißner

Medienkunde + Informatik = ? 57

Ira Diethelm, Christina Dörge

Zur Diskussion von Kontexten und Phänomenen in der Informatikdidaktik 67

Ira Diethelm, Christina Dörge, Ana-Maria Mesaros, Malte Dünnebier Die Didaktische Rekonstruktion für den Informatikunterricht	77
Ralf Romeike Informatiktools – Gestaltung einer Plattform für Werkzeuge für den Informatikunterricht	87
Dieter Engbring Was ist/kann/soll Informatikunterricht?	97
Beat Trachsler, Martin Guggisberg, Martin Lehmann Stereoskopische 3D-Videos selbst erstellen	107
Alexander Best Informatikgeschichte im Informatikunterricht – Konzepte und Materialien	117
Thiemo Leonhardt, Philipp Brauner, Jochen Siebert, Ulrik Schroeder Übertragbarkeit singulärer MINT-Interesse-initiierender außerschulischer Maßnahmen	127
Manuela Kalbitz, Hendrik Voss, Carsten Schulte Informatik begreifen – Zur Nutzung von Veranschaulichungen im Informatikunterricht	137
Michael Weigend Dramatisieren und literarisches Programmieren	147
Peter Antonitsch Kompetenzorientierung und Schulrealität	157
Dorothee Müller Fachdidaktisch begründete Auswahl von Informatiksystemen für den Unterrichtseinsatz	167
Ralf Romeike, Dominik Reichert PicoCrickets als Zugang zur Informatik in der Grundschule	177
Kerstin Strecker Zur Didaktik der Algorithmik	187
Ludger Humbert Schülerinnen konstruieren Informatische Bildung	197
Christian Wach Maschinelle Erfassung von Problemlösestrategien bei algorithmischen Problemstellungen am Beispiel des Sortierens	207

Simone Opel

Das Lernfeldkonzept in den Lehrplänen der IT-Berufe –
Vorstudie zur schülerseitigen Akzeptanz und Umsetzbarkeit
von selbstgesteuerten Lerneinheiten im Lernfeld
„Entwickeln und Bereitstellen von Anwendungssystemen“

217

Jan Schuster

Ein genetischer Zugang zum Programmieren mit CGI-Skripten in Python

227

Zehn Gründe Informatik zu studieren – Voraussetzungen, Motivation und Vorbereitung in der Schule

Stefan Jähnichen

Präsident der Gesellschaft für Informatik e.V. (GI)

TU Berlin

Fraunhofer FIRST

stefan.jaehnichen@gi.de

Es ist ein seltsames Phänomen der europäischen Industriestaaten: Seit der industriellen Revolution in Europa beruht die Blüte unserer Volkswirtschaften auf dem Erfindungsgeist und den konstruktiven Fähigkeiten unserer Ingenieure und trotzdem ist ihre gesellschaftliche Wertschätzung im Grunde genommen sehr niedrig – zu niedrig, um auf breiter Basis Begeisterung für diesen Berufsstand zu schaffen und ihn zumindest in der Nähe der Akzeptanz für Juristen, Mediziner oder Betriebswirte (= Manager in der Volksmeinung) zu bringen. Das ist nicht nur schade, sondern gefährdet den Wohlstand in Europa. Firmen, insbesondere die großen, international aufgestellten Konzerne, schaffen Arbeitsplätze dort, wo sie ausreichend ausgebildete Fachkräfte finden und auf ein Potential junger und kreativer Nachwuchskräfte treffen.

Wenn wir dies ändern wollen (eigentlich müssen wir das wollen !), müssen wir mehr junge Leute für diesen Berufsstand und speziell natürlich für den Innovationstreiber Nummer 1, die Informatik, motivieren und auch die bisher brachliegenden Potentiale heben: auch Mädchen frühzeitig für die Informatik zu interessieren.

Es ist klar, diese Begeisterung muss schon in der Schule – eigentlich aber sogar noch früher, im Elternhaus entwickelt werden und deshalb möchte ich meinen Vortrag dem Fach MINT und speziell natürlich – aus Freude an meinem Beruf – der Informatik widmen.

Was soll, was muss die Schule dazu leisten? Ich bin mir bewusst, dass die Diskussion darüber sehr weit auseinander geht. Die Schule soll das Medium Computer entmystifizieren – habe ich mal gehört –, die Schule soll den Umgang mit dem Computer lehren – also Word, Excel und Konsorten –, und die Schule soll den „sorgsamen“ Gebrauch des Internets vorbereiten – und sie soll den Jugendlichen eine Scheu vor dem Öffnen unliebsamer Seiten vermitteln.

Das ist ja alles richtig, aber geht doch eigentlich an der Sache vorbei! Die erstgenannten Themen können wir an den Schulen kaum mehr lehren ohne uns lächerlich zu machen und besonders die letztgenannten Themen sollten Pflichtbestandteil eines Sozialkunde – oder gar eines Ethikunterrichts sein, gegen den ich natürlich nichts einzuwenden habe – im Gegenteil! Aber es ist nicht der Stoff, mit dem wir Ingenieure erziehen oder gar für Informatik werben können.

Ich habe mich schon vor nunmehr gut dreißig Jahren für die Algorithmik in der Schule eingesetzt und bin heute mehr denn je der Überzeugung, dass dieser Teil der Informatik in die Schule gehört. Algorithmen entdecken, umsetzen, adaptieren und optimieren ist ebenso faszinierend wie es Mathematik sein kann – und das möchte ich Ihnen in meinem Vortrag vermitteln!

Die fehlenden neun Gründe, sich mit Informatik zu befassen oder gar zu studieren, werden natürlich auch vorgetragen.

Informatik: Kein Interesse?

Wolfgang Pohl

Bundeswettbewerb Informatik / BWINF

Wachsbleiche 7

53111 Bonn

pohl@bwinf.de

Abstract: Trotz relativ hoher Studienanfängerzahlen bleibt die Nachwuchssituation in der Informatik unbefriedigend. Eine Vielzahl an Maßnahmen arbeitet teilweise schon seit Jahren daran, das Interesse junger Menschen an Informatik zu wecken, es zu erhalten und das in Neigung und Begabung verwandelte Interesse zu fördern. Um noch effektiver und nachhaltiger wirken zu können, sollten die einzelnen Maßnahmen durch bessere Vernetzung untereinander und durch übergreifend wirkende Kommunikation gestärkt werden. Die rund um den Bundeswettbewerb Informatik entstandene Initiative „Bundesweit Informatiknachwuchs fördern“ (BWINF) unternimmt erste Schritte in diese Richtung und kann gleichzeitig als Bindeglied dienen.

Interessiert sich hier jemand für Informatik?

Die Klagen über einen Mangel an Fachkräften und an Nachwuchs für die Informatik- und IT-Wirtschaft reißen nicht ab. Dieses düstere Bild wird zumindest nicht von allen Statistiken nachgezeichnet: Die Zahl der Studienanfänger in Informatik (alle Hochschularten, Studienfächer einschließlich „Bindestrich-Informatiken“) erreichte laut Statistischem Bundesamt im Jahr 2010 einen neuen Rekordwert: beinahe 40.000 Anfänger in der Informatik, und die Tendenz der letzten Jahre weist nach oben. Das sind übrigens bereits im zweiten Jahr hintereinander mehr Anfänger als im Dot-Com-Boomjahr 2000, und man kann zur Zeit wirklich nicht von einem vergleichbaren IT-Hype sprechen. Der Anteil unter allen Studienanfängern ist praktisch stabil und liegt seit 2002 (also nach dem Platzen der Blase) irgendwo zwischen knapp 8,5 und gut 9 Prozent. (Alle Zahlen aus [StB10].)

Doch besonders an den Universitäten wird diese anscheinend positive Entwicklung nicht nachempfunden. Und hohe Anfängerzahlen schließen einen noch höheren Nachwuchsbedarf nicht aus. Neben den nackten Zahlen gibt es zudem qualitative Knackpunkte, die genauso bedeutsam sind. Zum einen bleibt die Frage nach weiblichem Nachwuchs in der Informatik akut: Der Frauenanteil unter den Anfängern ist zwar ebenfalls gestiegen und liegt nun bei knapp 20 Prozent, aber in Mathematik liegt er bei 50 Prozent; und selbst im Bauingenieurwesen gibt es einen höheren Frauenanteil als in der Informatik. Zum anderen gibt es Indikatoren, dass Jugendliche mit Informatik-naher Talentausbildung ein Studium in der Informatik eher vermeiden: Eine Studie der TU München zur Studienwahl zeigt,

dass eine gute Schulnote in Mathematik die Wahl von Informatik als Studienfach negativ beeinflusst, insbesondere im Vergleich mit informatiknahen Fächern [ELK08]. Dieses Fazit wird von Einzelfällen bestätigt: Unter den herausragenden Teilnehmern am Bundeswettbewerb Informatik sind immer wieder nennenswert viele, die ein Studium insbesondere in Mathematik oder Physik bevorzugen.

Wer sich über das Interesse von Jugendlichen an Informatik Gedanken macht, darf aber nicht nur die Situation an den Hochschulen in den Blick nehmen. Die Lage an den Schulen ist mindestens genau so wichtig. Thematisches Interesse wird wesentlich im Schulalter geprägt, hier muss sich Informatik gegen andere Wege „durchsetzen“. Und schließlich geht es nicht nur um Fachkräftenachwuchs, sondern mindestens genau so sehr um die allgemeine Verbreitung informatischer Bildung, die in einer derart informatisierten Gesellschaft wie der unseren immer noch weiter an Bedeutung gewinnt. Daran gemessen ist die Lage des Schulfachs Informatik prekär. Die an der TU Dresden im letzten Jahr unternommene Analyse zum Stand der informatischen Schulbildung [St10] zeigt für die gymnasiale Oberstufe, dass tendenziell sogar von einer Verringerung von Informatikunterricht gesprochen werden muss. Die Versuche, Informatik in übergreifenden MINT-Unterricht zu integrieren, scheitern häufig am Fehlen von entsprechend kompetenten Lehrkräften.

Angesichts der letztlich doch unbefriedigenden Situation, was die Entwicklung und Förderung von Interesse an Informatik angeht, bleibt genug zu tun. Einerseits gilt es, Informatik-Interesse zu wecken und jungen Menschen Anstöße zu geben, sich überhaupt mit Informatik zu beschäftigen. Andererseits muss das einmal angestachelte Interesse erhalten bleiben und über die evtl. schwierigen Phasen der Entwicklung Jugendlicher „hinübergerettet“ werden – Fördermaßnahmen müssen also nachhaltig sein.

Was wird getan?

Selbstverständlich ist die Nachwuchsfrage in der Informatik nicht neu. Dementsprechend gibt es viele Maßnahmen zur Nachwuchsförderung. Beginnen wir mit den „Anschubmaßnahmen“, bei denen es um erste Begegnungen mit Informatik geht. In den letzten Jahren spielt der noch junge Motivationswettbewerb „Informatik-Biber“ [PSH09] eine immer größere Rolle. Im Jahr 2010 nahmen knapp 120.000 Kinder und Jugendliche aus ca. 800 Schulen dieses Angebot wahr. Mit seinen Aufgaben versucht der Informatik-Biber einen erste, überraschende und weite Perspektive auf Inhalte der Informatik zu eröffnen. Ein attraktiver und fächerverbindender Ansatzpunkt ist die Beschäftigung mit einfachen Robotern. Hier gibt es z. B. die Wettbewerbe Robocup Junior und FIRST Lego League, die ebenfalls wachsende Teilnahmezahlen verzeichnen. Die Grundlagen für die Teilnahme werden vielfach an Schulen geschaffen, aber auch an außerschulischen Lernorten wie den „Roberta-Zentren“ – hier hat das Projekt Roberta der Fraunhofer-Gesellschaft das Entstehen einer wichtigen Infrastruktur bewirkt. Schwierig zu erfassen sind Aktivitäten im „Informatik-Streetworking“, also der Werbung für Informatik in der Öffentlichkeit. Singulär sind Hochschulen in diesem Bereich aktiv. Im Informatikjahr 2006 konnte das Projekt „Einstieg Informatik“ einige Module für öffentliche Informatik-Präsentationen zusammenstellen und in eigenen Präsentationen nutzen.

Ein Bereich mit Tradition ist die Talent- und Spitzenförderung. Schon 1980 wurde der Bundeswettbewerb Informatik ins Leben gerufen. Im Wettbewerb „Jugend forscht“ gibt es die kombinierte Sparte Mathematik/Informatik, die allerdings bei den Teilnehmerzahlen zu den schwächeren Bereichen dieser renommierten Fördermaßnahme gehört. Auf Bundeslandebene gibt es Schülerwettbewerbe in Sachsen und Brandenburg, neuerdings auch in Mecklenburg-Vorpommern. Viele Hochschulen bieten besonders begabten Schülerinnen und Schülern die Möglichkeit zu einem Schülerstudium in Informatik an. Hier überlappt sich Begabtenförderung mit Studienwerbung. In diesem Bereich geht es darum, Schülerinnen und Schüler für ein Studium im eigenen Fach zu gewinnen – und möglichst auch an der eigenen Hochschule. Im gleichen Schnittgebiet sind Schülerworkshops einzuordnen, während (allgemeine oder fachbezogene) Schnuppertage oder ähnliche Veranstaltungen eher reine Studienwerbung sind.

Schülerworkshops, ob an Hochschulen oder anderen Lernorten, und andere außerschulische Lernangebote sind dazu geeignet, Interesse zu erhalten und zu bestärken. Vorbildlich ist hier das Schülerrechenzentrum der TU Dresden, dessen Kurssystem ein in Deutschland wohl einmaliges Förderangebot im Bereich der Informatik darstellt. Tradition hat auch das Informatik-Sommercamp der Universität Passau, das in diesem Jahr bereits zum 16. Mal durchgeführt wurde und unter interessierten Schülerinnen und Schülern bundesweite Bekanntheit erlangt hat. Das Projekt „Einstieg Informatik“ bietet auch lange nach Abschluss des Informatikjahrs ein Webportal für interessierte Jugendliche an, das in 2009 mit Unterstützung des Fakultätentags Informatik erneuert wurde, seitdem gemeinsam mit über 20 Hochschulen betrieben wird und u.a. einen Community-Bereich enthält.¹

Was bleibt zu tun?

Die Vielzahl der zum Teil schon seit vielen Jahren bestehenden Initiativen und Angebote hat viel Positives bewirkt, aber die anfangs geschilderte Nachwuchsproblematik nicht verhindern können. Wo sind vielleicht noch Schwachpunkte, was lässt sich verbessern?

Die immer noch zu schwache und instabile Situation des Schulfachs Informatik muss zuvorderst auf schulischer Ebene verbessert werden. Eine gleichzeitige, möglichst prominente außerschulische Nachwuchsförderung in diesem Gebiet kann dies nur unterstützen. Die rund um den Bundeswettbewerb entstandene Initiative „Bundesweit Informatiknachwuchs fördern“ (BWINF²) bemüht sich, geeignete Wege zu probieren und aufzuzeigen. Die einzelnen Projekte – außer dem Bundeswettbewerb sind das der Informatik-Biber, Einstieg Informatik und das Auswahlverfahren zur Internationalen Informatikolympiade – werden einerseits besser vernetzt und andererseits mit der gemeinsamen Marke BWINF versehen. BWINF-intern bedeutet Vernetzung u.a.: Mit dem in der Breite wirkenden Informatik-Biber kann auch der Bundeswettbewerb besser bekannt gemacht werden; die geringe Einstiegsschwelle des Informatik-Biber verhilft Jugendlichen und Lehrkräften zur Berührung mit Wettbewerbsangeboten und senkt so gelegentlich die Hemmungen vor einer Teil-

¹ www.einstieg-informatik.de

² www.bwinf.de

nahme bzw. Unterstützung des Bundeswettbewerbs. Einstieg Informatik ist als Interesse-erhaltendes Angebot zwischen den beiden Wettbewerben positioniert. Schließlich können die an der Olympiade-Auswahl teilnehmenden Spitzentalente als Vorbild für Jüngere dienen – wenn sie denn nicht nach ihrer Olympiade-Karriere Mathematik studieren wollen. Einen Überblick über die BWINF-Projekte gibt Abbildung 1.

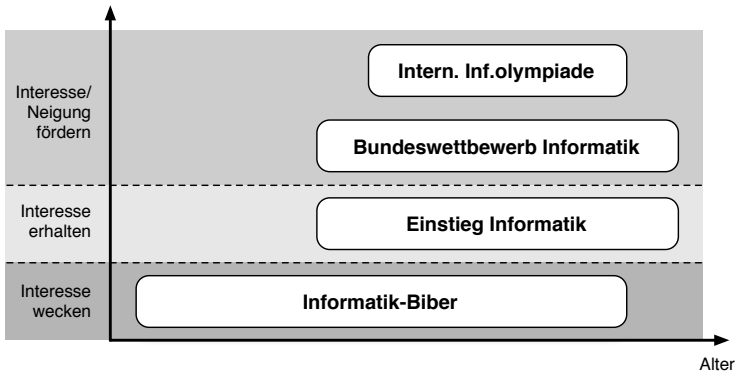


Abbildung 1: BWINF-Projekte, ihre Wirkung auf Informatik-Interesse und ihre Altersreichweite.

Vernetzung nach außen war das wesentliche Ziel einer BWINF-Sonderaktion im letzten Jahr, bei der die Teilnahme von Mädchen an Informatik-Workshops (meist von Hochschulen veranstaltet) gefördert wurde. Im Gegenzug bekam BWINF Gelegenheit, bei diesen Veranstaltungen die eigenen Projekte zu bewerben. Ein weiteres Beispiel: Die beim BWINF-Portal Einstieg Informatik engagierten Hochschulen können ihre Angebote dort kommunizieren und erreichen damit auch Teilnehmer der BWINF-Wettbewerbe. Auch die von verschiedenen Hochschulen ausgerichteten Workshops für Teilnehmerinnen und Teilnehmer am Bundeswettbewerb sind Beispiele für Vernetzung, genauso wie die Teilnahme des Schülerrechenzentrums Dresden an Informatik-Biber und Bundeswettbewerb.

Die Wirkung einer starken Marke ist bekanntlich nicht zu unterschätzen. Im Bereich der Wirtschaft haben Marken teilweise einen immensen Wert. Im Bereich der Nachwuchsförderung zeigt Roberta, wie eine Marke ein Gebiet (hier die „educational robotics“) voranbringen kann. Die Auszeichnung von im Bundeswettbewerb besonders aktiven Schulen als „BWINF-Schule“ ist ein erster Schritt, die Marke BWINF als Kennzeichen aktiver Nachwuchsförderung einzusetzen.

Ein vermutlich für die Nachwuchsförderung sehr wichtiges Ziel ist, die Wahrnehmung und Vorstellungen von Informatik bzw. Informatikerinnen und Informatikern zu verbessern. Ein kleiner Schritt in diese Richtung ist der „Köpfe“-Bereich des Portals Einstieg Informatik. Hier werden bekannte und weniger bekannte Informatiker – und bewusst überproportional Informatikerinnen – dem jugendlichen Zielpublikum auf persönliche Weise näher gebracht und gleichzeitig die Vielfalt an Themen und Arbeitsbereichen der Informatik vorgestellt. Im Bereich des „Image-Building“ für Informatik bleibt allerdings noch besonders viel zu tun.

Literaturverzeichnis

- [ELK08] Stefan Engeser, Nina Limbert und Hugo Kehr. Abschlussbericht zur Untersuchung „Studienwahl Informatik“ München, Juli 2008. Technische Universität München.
- [PSH09] Wolfgang Pohl, Kirsten Schlüter und Hans-Werner Hein. Informatik-Biber: Informatik-Einstieg und mehr. In Bernhard Koerber, Hrsg., *Zukunft braucht Herkunft*, Lecture Notes in Informatics, Seiten 38–49, Bonn, 2009. Gesellschaft für Informatik.
- [St10] Isabelle Starruß. Analyse der informatischen Bildung an allgemein bildenden Schulen auf der Basis der im Jahr 2010 gültigen Lehrpläne und Richtlinien. Dresden, 2010. TU Dresden, Didaktik der Informatik. <http://dil.inf.tu-dresden.de/Synopse-zum-Informatikunterricht-in-Deutschland.290.0.html>.
- [StB10] Bildung und Kultur. Schnellmeldungsergebnisse der Hochschulstatistik zu Studierenden und Studienanfänger/-innen – vorläufige Ergebnisse. Wiesbaden, 2010. Statistisches Bundesamt.

Informatik für die Welt von Morgen

Andreas Schreiber

Simulations- und Softwaretechnik
Deutsches Zentrum für Luft- und Raumfahrt e.V.
Linder Höhe
51147 Köln
Andreas.Schreiber@dlr.de

Im Deutschen Zentrum für Luft- und Raumfahrt wird auf den Gebieten Luftfahrt, Raumfahrt, Energie, Verkehr und Sicherheit geforscht und entwickelt. Das umfasst die Erforschung von Erde und Sonnensystem, Forschung für den Erhalt der Umwelt und umweltverträgliche Technologien, zur Steigerung der Mobilität sowie für Kommunikation und Sicherheit. Damit arbeitet das DLR auf vielen Gebieten, die unsere Welt von Morgen entscheidend prägen und verändern werden.

In allen Bereichen des DLR ist Informatik zu einer entscheidenden Technologie geworden. In fast allen Instituten und Einrichtungen wird Software entwickelt. Insgesamt entwickeln ca. 1000 der etwa 6500 DLR-Mitarbeiter Software. Dabei handelt es sich überwiegend um Individualsoftware. Dies ist notwendig, da oft keine Standardsoftware am Markt verfügbar ist, welche die Anforderungen des DLR erfüllt. Oder die existierende Standardsoftware besitzt nicht die ausreichende Qualität oder Genauigkeit, so dass eine Eigenentwicklung gewünscht ist.

Die Art der im DLR entwickelten Software lässt sich grob unterteilen in:

- Software mit hoher Kritikalität. Hierzu zählen vor allem echtzeitfähige Software und Software für eingebettete Systeme. Diese Software hat oft hohe Anforderungen an Ausfallsicherheit und Fehlerfreiheit. Sie steuert oft technische Systeme und häufig ist das Leben von Menschen von ihr abhängig.
- Simulationssoftware. Diese Art Software dient zur Simulation physikalischer Vorgänge oder komplexer Systeme. Sie hat oft hohe Anforderungen an Genauigkeit und Performanz. Dementsprechend werden oft große Datenmengen erzeugt und die Simulationssoftware wird auf High-Performance-Computing-Rechnern ausgeführt. Ein typisches Beispiel aus dem DLR ist Software für numerische Strömungssimulation.
- Unterstützende Software. Darunter versteht man Software, welche die Arbeit von Wissenschaftlern unterstützt und produktiver macht. Hierzu zählt Software zum Verwalten von Daten, zum Dokumentieren und Nutzbarmachen von Wissen oder zur (graphischen) Auswertung von Daten.

- Administrative Software und Software für große Nutzerzahlen. Das ist häufig Web-basierte Software für Internet oder Intranet. Oft mit Anbindung an Unternehmenssoftware wie SAP. Beispiele sind Workflow-Anwendungen zum beantragen von Reisen, Urlaub oder Beschaffungen oder Software zur verwalten von IT-Ressourcen.

Zum Entwickeln der Software werden viele verschiedenen Software-Technologien eingesetzt. Dazu gehören Programmiersprachen, von denen Python, C++ und Java am häufigsten eingesetzt werden. Allerdings werden im gesamten DLR über 30 unterschiedliche Programmiersprachen genutzt. Oft haben diese jedoch eng begrenzte spezielle Anwendungsgebiete. Es werden auch viele unterschiedliche Software-Frameworks, Bibliotheken und Module verwendet. Ein bekanntes Beispiel dafür ist das Eclipse-Framework. Zur Entwicklung werden aktuelle Architekturen verwendet, z.B. komponentenbasierte Architekturen.

Sehr wichtig ist im DLR das Software Engineering. Für alle Software-Entwicklungen sind ein vernünftiger Software-Entwicklungsprozesse und geeignete Entwicklungswerkzeuge notwendig, um Software von hoher Qualität zu erstellen. Das DLR erforscht daher, welche Entwicklungsprozesse und Werkzeuge sich gut für Wissenschaftler und Ingenieure eignen. Diese entwickeln in erheblichen Umfang Software als Teil ihrer Arbeit. Jedoch ist Software-Entwicklung ist nicht ihr Hauptinteresse sondern lediglich Mittel zum Zweck, um ingenieur- oder naturwissenschaftliche Aufgaben zu bearbeiten. Daher müssen Software-Engineering-Methoden einerseits ausreichen um die Qualität der Software zu erhöhen, aber dürfen andererseits die Ingenieure nicht in ihrer Arbeit behindern.

Software wird im DLR meist in interdisziplinären Teams entwickelt. Ingenieure, Mathematiker, Physiker, Chemiker oder Mediziner entwickeln gemeinsam mit Informatikern anspruchsvolle Software. Das ist für Informatiker meist ein sehr spannendes Umfeld, in dem sie viele Einblicke in die verschiedenen Fachdisziplinen bekommen. Allerdings ist auch gute Kommunikation in Entwicklungsprojekten notwendig für erfolgreiche Resultate. Interesse, Lust und Spass am kommunizieren mit „fremden“ Fachdisziplinen ist daher sehr wichtig.

Das DLR ist sehr interessiert an der Förderung des Nachwuchses. Speziell für Schüler gibt es zwei Aktivitäten, die Interesse an ingenieur- und naturwissenschaftlichen Themen wecken sollen: DLR_School_Labs und DLR_next. Schwerpunkt sind dabei, passend zum DLR, Themen aus Raumfahrt, Luftfahrt, Energie und Verkehr.

Die DLR_School_Labs sind die Schülerlabore des DLR. In den DLR_School_Labs entdecken Schüler aktiv die Welt der Forschung und Technik. Sie können direkt in verschiedenen Standorten des DLR Experimente durchführen und so spielerisch erfahren, wie spannend Naturwissenschaften und Forschung sein können.

DLR_next ist das DLR-Jugendportal im Internet. Es bietet Jugendlichen, Schülern und Kindern Informationen und Multimedia-Angebote rund um die Forschungsthemen des DLR.

Agil und spielerisch: Neue Methoden der Software-Entwicklung in der Praxis und ihr Potential für den Schulunterricht

Dr. Stefanie Scherzinger

Google Germany GmbH
Dienerstraße 12
80331 München
steffis@google.com

„Agile Software-Entwicklung“ bezeichnet eine iterative Vorgehensweise, die sowohl in Silicon Valley Startups als auch großen Software-Häusern erfolgreich eingesetzt wird. Musste nach dem Wasserfall-Modell jede Eventualität im Voraus akribisch bedacht werden, sind agile Ansätze deutlich flexibler: In kurzen Zyklen wird eine einfache, prototypische Lösung zum vollwertigen Produkt ausgebaut. Am Ende jedes Zyklus steht das fertig getestete Produkt; auch wenn es möglicherweise noch nicht über sämtliche gewünschten Funktionalitäten verfügen sollte, ist es für die wichtigsten Zwecke bereits einsatzbereit. Somit können Auftraggeber zeitnah Änderungswünsche einbringen und das Produkt noch in der Entstehungsphase auf das Nutzungsziel ausrichten. Das Entwicklungsteam wiederum kann auf Unvorhergesehenes flexibel reagieren und Zwischenerfolge entsprechend verbuchen. Die kritische Retrospektive spielt dabei eine tragende Rolle.

Methoden wie Agiles Planen, Scrum Meetings und Test Driven Development zählen zu den festen Lehrinhalten an Hochschulen. Dieser Vortrag zeigt den Fortschritt durch die agile Software-Entwicklung auf. Bei der Darstellung der neuen Konzepte wird vor allem auf deren Praxisorientierung hingewiesen. So betrachten wir Beispiele aus dem Arbeitsalltag einer Software Entwicklerin und diskutieren, wie diese Konzepte auch im schulischen Informatikunterricht spielerisch erfahrbar gemacht werden können:

- Bei der Projektplanung spielen Zeitabschätzungen eine wesentliche Rolle. Im Kartenspiel “Planning Poker” pokert das Team um die Aufwände einzelner Funktionalitäten.
- In täglichen Lagebesprechungen, den Scrum Meetings, moderiert ein Scrum Master gezielt den Informationsfluss. Mit Antworten auf einfache Fragen (was habe ich gestern erledigt, woran arbeite ich heute, und wo „hakt es“ bei meiner Arbeit?) wird der Stand des Projekts für das Team erkennbar. Spielerisch kann ein solcher Ablauf am Beispiel gemeinschaftlichen Kochens geübt werden.

- Test Driven Development ist ein bewährtes Prinzip, bei dem erst die Tests und dann der eigentliche funktionale Teil des Projekts programmiert werden. So wird sichergestellt, dass das Programm auch gut zu testen ist. Im Gruppenspiel "Spaghetti Challenge" bauen kleine Teams in kürzester Zeit Türme aus Spaghettis, Bindfaden und Klebeband. Es gewinnt das Team, dessen Turm ein Stück Marshmallow auf dem höchsten Punkt balancieren kann. Doch Vorsicht, wer das scheinbar leichte Marshmallow-Stück erst zuletzt auf den Turm setzt, überschätzt in der Regel die Tragkraft der gebauten Struktur, - wie auch bei der Software Entwicklung die Belastungstests gerne unterschätzt werden.
- Vier Augen sehen mehr als zwei. Nach diesem Prinzip funktionieren Code Reviews, bei denen alle Änderungen am Programm erst von einer zweiten Person akzeptiert werden müssen, bevor sie umgesetzt werden dürfen. Im Pair Programming wird gar zu zweit am Rechner entwickelt. Was zunächst als Einschränkung der Kreativität und als Kontrolle empfunden werden könnte, wirkt in der Praxis aber sehr effektiv. Sobald das Ziel gemeinsam erarbeitet ist, kommt auch der Spaß an der Arbeit nicht zu kurz.

Projektmanagement und die Fähigkeit zur Selbstorganisation sind wichtige Schlüsselkompetenzen in der Berufspraxis. Indem Schüler und Schülerinnen erste Erfahrungen in der Software Entwicklung sammeln, erhalten sie Einblick in die Berufsbilder von Software-Entwicklung und Projektleitung. Unter Betonung der kommunikativen Komponenten der Programmierung möchte der Vortrag nicht nur häufigen Vorbehalten gegenüber der Informatik, wie das negative Image „des sozial vereinsamten Hackers“, entgegen wirken, sondern auch neues Interesse wecken. Zudem erhalten Schüler und Schülerinnen Handwerkszeug für ihr eigenes Zeitmanagement, wie es für die Persönlichkeitsentwicklung wertvoll ist.

Die Referentin greift bei ihren Vorschlägen für den Schulunterricht auf ihren Erfahrungsschatz als Software Entwicklerin bei IBM und Google zurück, sowie auf ihr nebenberufliches Engagement in der fachspezifischen Didaktik. Dazu gehören die Gestaltung von Mädchen-Techniktagen oder Veranstaltungen bei der Informatik-Sommerschule "Informatica Feminale", die sich an Informatikerinnen in Studium und Beruf wenden.

Die Medien der Informatik

Jochen Koubek

Digitale Medien
Universität Bayreuth

jochen.koubek@uni-bayreuth.de

Das Verhältnis zwischen Informatik und Medienwissenschaft, insbesondere zwischen Informatikdidaktik und Medienpädagogik ist traditionell ziemlich angespannt. Medienwissenschaftler werden von Informatikern gerne als ‚Schmalspurfilmer‘ diskreditiert, um klarzustellen, dass eine Beschäftigung mit Medien innerhalb der Informatik wenig gehaltvoll und nicht ernst zu nehmen ist. Informatiker hingegen werden in den Medienwissenschaften vor allem als Ingenieure wahrgenommen, die sich zwar famos mit Technik auskennen, zu Fragen von mediengesellschaftlicher Relevanz aber wenig beizusteuern haben.

Derartige Attributierungen haben für soziale Formationen eine identitätsstiftende und -stabilisierende Funktion, dienen sie doch der Innen-/Außendifferenzierungen und markieren die Grenzen dessen, was die Gruppe als wesentlich empfindet. Von außen gesehen sind sie vor allem amüsant und wenn es keine ernsthaften Berührungspunkte zwischen den Gruppen gibt, bleiben sie ohne Konsequenzen.

Sobald die Interessen der Gruppen sich aber zu überschneiden beginnen, ist es ratsam, alte Feindbilder zu überdenken und Gemeinsamkeiten und Unterschiede neu abzustechen. Im Fall Informatik und Medien bedeutet dies:

- Informatik ist im Alltag allgegenwärtig.
- Informatische Bildung (in welcher Form auch immer) ist unverzichtbar für die Bewältigung des Alltags.
- Dennoch steckt der Informatikunterricht in einer Legitimationspflicht gegenüber der Gesellschaft und ihren politischen Vertretern.
- Eine erfolgversprechende Legitimationsstrategie für die Informatik ist der Verweis auf die Bedeutung ihrer Grundlagen, z.B. auf Anwendungskontexte.
- Wenn die Informatik auf ihre Anwendbarkeit hinweist, deutet sie regelmäßig auf Medien.

- Für die Gesellschaft – Schüler, Eltern, Politiker, Journalisten – zeigt sich Informatik sogar primär in digitalen Medien, z.B. Handys, SMS, WWW, Email, Foren, IMS, Youtube, Twitter, Facebook, Computerspiele.
- Die am heftigsten geführten gesellschaftlichen Aushandlungsprozesse um Informatische Systeme haben Digitale Medien im Zentrum: Urheberrecht und Geistiges Eigentum, Privatheit und Überwachung, Informationsfreiheit und Zensur, Jugendschutz und Persönlichkeitsrechte.
- Bildungspolitiker verstehen eher die Notwendigkeit von Medienbildung als die Notwendigkeit von informatischer Bildung.
- Es ist daher eher ein Pflichtfach »Medien« abzusehen als ein Pflichtfach »Informatik«.
- Die Medienpädagogik arbeitet seit vielen Jahren an Konzepten, wie und in welcher Form Medienbildung im Bildungssystem verankert werden kann.
- Wenn die Informatik bei der Gestaltung mitwirken möchte, tut sie gut daran, Medien als Teil ihres Wirkungskreises zu akzeptieren und entsprechende Konzepte aber auch Etikettierungen anzubieten.

Im Vortrag wird das Spannungsfeld zwischen Informatik und Medienwissenschaft entfaltet und Strategien für eine fruchtbare Zusammenarbeit angesprochen.

Visuelle Programmierung - oder: Was lernt man aus Syntaxfehlern?

Eckart Modrow

Didaktik der Informatik
Universität Göttingen
Goldschmidtstrasse 7
37077 Göttingen
emodrow@gmx.de

Abstract: Visuelle Entwicklungsumgebungen gibt es schon lange, weil sich durch die Vermeidung von Syntaxfehlern einerseits Entwicklungszeiten verringern lassen, andererseits Programmieranfängern die Möglichkeit gegeben wird, sich schnell auf die inhaltlichen Aspekte des Programmierens zu konzentrieren. Allerdings waren diese Systeme meist auf enge Anwendungsbereiche beschränkt. Mit neuen, sehr viel universeller einsetzbaren Werkzeugen wie BYOB¹ lassen sich alle Bereiche der Schulinformatik oder einer Anfängervorlesung abdecken. Damit stellt sich die Frage, ob sie nicht in absehbarer Zeit die klassischen textbasierten Sprachen ablösen können – und welche Konsequenzen das hätte. Der Beitrag beruht auf Unterrichtserfahrungen mit visuellen Entwicklungsumgebungen in den Klassenstufen 5 bis 13 sowie Vorlesungen für Programmieranfänger in den Naturwissenschaften.

1 Einleitung

In der Informatik gibt es zwei weitgehend unbestrittene Aussagen zum Bereich Algorithmen/Programmieren: 1) „Wichtig sind die Algorithmen, nicht deren Codierung.“ und 2) „Richtiges Programmieren ist textbasiert“. Lange Zeit konnte nur textbasiert programmiert werden. Es wurde deshalb auch kaum hinterfragt, welche Bedeutung die Codierung selbst für die Allgemeinbildung haben könnte, denn wenn die Implementierung von Algorithmen einen Bildungswert besitzt, dann war die Codierung nicht zu vermeiden. Leider ist das so wenig angesehene Codieren aber ausgerechnet die Haupthürde für Programmieranfänger.

¹ [MH10]

Fordert man, dass nach Abschluss eines wie auch immer gearteten Programmierlehrgangs eine Programmiersprache als Werkzeug zur Formulierung und Implementierung eigener Ideen und Problemlösungen genutzt werden kann, dann scheitert der überwiegende Teil der Anfänger, bis zu 80%, an dieser Hürde sowohl im Informatikunterricht der Schulen² wie auch an den Hochschulen. Man muss das wohl fordern, denn das reine Lesen und Nachvollziehen fertiger Programmtexte erscheint sinnlos, weil Algorithmen genauso gut, meist besser, in anderen Notationsformen formuliert und vermittelbar sind – und auf die kommt es ja angeblich nur an. Es ist also kein Zufall, dass auf vielen Ebenen versucht wird, diesen Zustand zu verbessern. Reduzierte Modellsysteme mit eingeschränkten Befehlssätzen, Programmgeneratoren, vereinfachte Sprachen wurden und werden eingeführt – und eben visuelle Entwicklungssysteme. Deren Nutzung wollen wir etwas genauer betrachten.

2 Zur Akzeptanz visueller Entwicklungssysteme

Visuelle Entwicklungsumgebungen gibt es sehr unterschiedlicher Art: als Vertreter der Zustandsmodellierung mögen die Komponente AutoEdit aus AtoCC, Kara oder der schöne USBomat³ von Helmar Becker dienen. Sie steuern ebenso wie blockorientierte Systeme den Kontrollfluss, aber diese kommen der traditionellen Programmierung sehr viel näher, egal ob sie struktogrammartige Strukturen wie die erste LEGO-Mindstorms-Version und Scratch oder Flussdiagramme benutzen, wie sie besonders im technischen Bereich verbreitet sind. Sie scheinen damit auch geeigneter, auf die textbasierte Programmierung vorzubereiten – wenn das denn erforderlich ist. Etwas aus dem Rahmen fallen Systeme wie LabView und die daraus abgeleiteten neueren LEGO-NXT-Systeme, die eher datenflussorientiert arbeiten, sowie Sonderwege wie das Smalltalk-basierte eToys oder das 3D-Entwicklungssystem Alice⁴, das die Erfolgsquote an der Carnegie-Mellon-Universität bei den Programmieranfängern in etwa verdoppelt hat. Dass es sich nicht um Spielzeuge handeln muss, zeigt die Steuerung des südafrikanischen Großteleskops SALT, die komplett über LabView erfolgt. Entsprechendes lässt sich über die Aktivitätsdiagramme von UML sagen, die ebenfalls für den professionellen Einsatz gedacht sind und dort auch zunehmend Verwendung finden.

Man soll nur nicht glauben, dass neue Entwicklungen immer auf ungeteilte Begeisterung stoßen! Neue Techniken entwerfen altes Expertenwissen – und das gefällt meist nur den Laien; die Experten finden das gar nicht witzig. Neu ist diese Beobachtung nicht: schon der Übergang von Assemblern zu universellen Programmiersprachen wurde kritisch begrüßt, und ältere Kolleginnen und Kollegen werden sich an den nicht nur ironisch gemeinten Aufsatz „Echte Programmierer meiden Pascal“ erinnern, der das Aufkommen der strukturierten Programmierung als „Müslifresser-Programmierung“ kommentierte⁵. Folgerichtig gehören die GUI-Builder wie Delphi, NetBeans oder Eclipse in den Bereich des „Klicki-Bunti“.

² z. B. in [Hu00]

³ [HW04], [NHR99], [Be08]

⁴ [MI07], z. B. [Fi10], [CM99]

⁵ z. B. in [Le83]

Wir können also nicht erwarten, dass die Abkoppelung der Implementierung schultypischer Algorithmen vom Gebrauch universeller Programmiersprachen von den Experten begrüßt wird. Wird sie auch nicht. Eine hübsche Reaktion eines hier ungenannt bleibenden Kollegen auf die Vorstellung von BYOB war: „Lernen muss weh tun!“ Besser kann man es nicht ausdrücken.

Was sagen nun die Betroffenen dazu?

Erstaunlicherweise sehen sie das sehr ähnlich. Programmieranfänger aus den Naturwissenschaften des dritten Semesters, die sowohl Java wie Scratch⁶ kennen gelernt hatten, äußerten überwiegend⁷, dass Scratch für den Einstieg außerordentlich wichtig gewesen wäre (m: 83%, w: 100%) und auch weiterhin parallel zu Java eingesetzt werden sollte, weil es für das Verständnis der Strukturen und Abläufe wichtig sei (m: 67%, w: 100%), außerdem mache es viel Spaß (m: 67%, w: 83%). Trotzdem sei es nicht „richtiges Programmieren“ (m: 33%, w: 67%), es fehlten wichtige Strukturen (m: 83%, w: 50%) und sei nur für den Anfang geeignet (m: 50%, w: 33%). Ihre persönliche Präferenz liege nach der Veranstaltung bei Java (m: 100%, w: 50%).

Dazu einige Zitate:

„Nach dem Arbeiten mit Java stellt sich allerdings das Gefühl ein, dass man mit Scratch nicht viel wirkliches Programmieren lernt. Man lernt bei Scratch weder etwas über Klassen oder Methoden, noch hinterblickt man die wirkliche Strukturierung eines Programms.“

„Scratch eignet sich ... um ...ein Verständnis für Algorithmik zu bekommen. ...Dennoch empfinde ich es als wichtig, auch Java zu lernen, da viele Programme ...damit geschrieben sind und es somit beim Verständnis von diesen hilft.“

„Ich finde Scratch gut, um die Denkweise ...und den Aufbau eines Programmes kennenzulernen. Die Programmiersprache Java ist sehr wichtig, da mit ihr die meisten Programme geschrieben sind“

„(Zu Scratch:) Man versteht ziemlich schnell die Grundgedanken, die man fürs Programmieren braucht und alles ist sehr übersichtlich. Zudem macht Scratch es einem fast unmöglich Fehler zu machen. ... (Zu Java:) Zudem ist es wahnsinnig frustrierend, wenn man fast alles richtig gemacht hat, das Programm aber dennoch nicht läuft, weil sich irgendwo ein kleiner Fehler eingeschlichen hat., aber danach ...muss auch Java gezeigt werden, da es doch sehr viel mehr Möglichkeiten bietet.“

„Ich denke, dass Scratch ... auch nicht ... in Frage kommt, da es meiner Meinung nach für eine Präsentation durch den Lehrer äußerst unseriös wirkt.“

Fazit: Java is the real thing!

⁶ BYOB war im Sommersemester 2010 noch in der Entwicklung.

⁷ Es handelt sich um 12 Interviews (m:6, w:6), die typisch für die Reaktion der Teilnehmenden aus zwei Veranstaltungen waren. Die Zahlen sollen nur eine Tendenz illustrieren.

Die Schülerinnen und Schüler sehen das genauso. Einerseits arbeiten sie sehr gerne und erfolgreich mit Scratch, andererseits wollen sie nach einer Weile „richtiges Programmieren“ kennen lernen. Eine in beiden Systemen relativ erfahrene Schülergruppe der Klassenstufe 11 schilderte sehr eindringlich, dass sie z. B. bei der Einführung von Variablen und einfachen Datentypen mit Java das Gefühl hätten, der Maschine „viel näher“ zu sein und sich deshalb auch viel sicherer fühlten als bei visueller Programmierung. Sie schlugen sogar vor, aus diesem Grund mit Java zu beginnen (!) und danach Scratch für die „richtigen Arbeiten“ zu benutzen, weil das viel effizienter sei. Das Gefühl, etwas gut verstanden zu haben, ist entscheidend für den Lernerfolg. Schon deshalb sind diese Äußerungen sehr ernst zu nehmen. Die Einschätzungen ähneln der Diskussion in der Linux-Gemeinde zwischen den Anhängern der Konsolennutzung und den Freunden der grafischen Oberflächen. Beides sind „Shells“ unterschiedlichen Aussehens, aber gleicher Funktionalität – mehr nicht. Man mag das eine mögen oder das andere. Größere Unterschiede gibt es nicht, schon gar nicht in der „Nähe“ zur Maschine.

Betrachten wir einmal die sachlichen Grundlagen der Aussagen. Die entwickelten Java-Programme waren weder hinsichtlich ihrer Funktionalität noch in den algorithmischen Strukturen komplexer als ihre Scratch-Pendants – eher deutlich umgekehrt. Das hat die Schülergruppe auch richtig erkannt. Der Aufwand für ihre Erstellung war dagegen wesentlich umfangreicher. Die benutzten Datentypen waren in beiden Systemen vorhanden, Anfänger benötigen nun mal keine anspruchsvollen Strukturen. Das Argument, dass sehr viel Software in Java erstellt wurde, zählt natürlich nicht für die Anwender – denen kann die Entwicklungsumgebung egal sein. Da es keine Anschlussveranstaltung für die Studierenden gab, werden fast alle auch keine umfangreicheren Programme mehr schreiben. Wir müssen also wohl akzeptieren, dass Programmieranfänger, obwohl sie erfahren haben, dass z. B. Scratch für entsprechend gewählte Beispiele einfacher, verständlicher und fehlertoleranter, auch im Gebrauch viel schneller ist als ein Java-Entwicklungssystem, Scheinargumente heranziehen, um die textbasierte Programmierung zu rechtfertigen. Sie wollen textbasiert arbeiten – unabhängig von dessen Sinn. Wenn das so ist, dann haben wir ein echtes Lernhindernis, das die Vorteile der visuellen Programmierung völlig kompensieren kann. Können wir es überwinden? Sollen wir es überhaupt überwinden?

Ziehen wir eine erste Zwischenbilanz:

1. Wenn sich die Lernenden algorithmisches Problemlösen und die dafür erforderlichen Methoden und Erfahrungen aneignen sollen, dann kommt es nach einhelliger Auffassung auf die Algorithmik an, nicht auf die Codierung – und genau dafür sind visuelle Entwicklungssysteme wie Scratch/BYOB die geeigneteren Werkzeuge, zumindest in einem Anfängerkurs.
2. Praktisch alle Beteiligten sind sich gefühlsmäßig darüber einig, dass es „irgendwie doch“ auf die Codierung ankommt, die Experten vielleicht, weil sie ihr Expertentum gefährdet sehen oder schlicht aus Gewohnheit, die Lernenden vielleicht, weil sie Experten nach den vorliegenden Kriterien werden wollen, also Experten im Codieren.

3. Die langjährigen Erfahrungen zeigen, dass das Codieren zu einer eigentlich völlig inakzeptablen Misserfolgsquote führt, die aber trotzdem akzeptiert wird, oft sogar gewollt ist. Die neuen Möglichkeiten, diese Situation zu verbessern und zu denen wir noch kommen, werden von beiden Seiten kaum angenommen – beide Seiten wollen sie eigentlich nicht.
4. Die vorhandenen Vorbehalte, die wir allerdings auch schon in der Vergangenheit bei strukturellen Änderungen bei den Entwicklungssystemen vorfanden, mögen ihre Berechtigung haben. Diese ist bisher aber überhaupt nicht untersucht, insbesondere ist der eigenständige allgemeinbildende Wert textbasierter Codierung bisher nicht begründet, weil keinerlei Notwendigkeit für eine Begründung vorlag. Die Codierung hat sich ihre Bedeutung einfach von der Algorithmik, mit der sie – bisher – unzertrennbar verknüpft war, entliehen.

Vergleichen wir die Situation einmal mit der Motivierung für die Naturwissenschaften. Schülerinnen und Schüler werden in ein Schülerlabor gekarrt, einen außerschulischen Lernort, extrahieren dort Farbstoffe aus Gemüseteilen und tragen dabei einen weißen Kittel und – natürlich – eine Schutzbrille, obwohl die gar nicht nötig ist. Sie fühlen sich dadurch aber als kleine Chemiker, es kommt nicht auf das Extraktionsverfahren an, sondern auf die Schutzbrille, die äußeren Insignien des Wissenschaftlers. Die motivierende Wirkung der Aktion ist deutlich spürbar.

Vielleicht spielen die textbasierten Programmiersprachen eine ähnliche Rolle wie die Schutzbrille als eine Art Geheimschrift, die niemand lesen kann außer den Experten. Wenn dem so sein sollte, dann kommen wir allerdings mit rationalen Argumenten nicht mehr weiter. Wir hätten es mit Symbolen zu tun, mit Bildern für einen Berufsweig. Wenn kryptische Geheimschrift das Symbol für die Informatik ist, was – nebenbei – ziemlicher Unsinn wäre, dann wäre die Ansicht der Lernenden verständlich, dass die Beherrschung dieser Geheimschrift zum Initiationsritus der Informatik gehört, dass das Erlernen von z. B. Java unabhängig von dessen Sinnhaftigkeit den Informatiker kennzeichnet, dass erst das „richtige“ Informatik wäre. Weiterhin erklärte es auch wenigstens teilweise die Ablehnung des Faches in weiten Kreisen. Geheimschriftexperten sind Sonderlinge, grenzen sich durch ihre Geheimnistuerei ab, liefern kein positives Bild für überwiegend offen und optimistisch in die Zukunft blickende Jugendliche.

Die Wertschätzung der textuellen Codierung besonders im Schulbereich liefert ein Bild des Faches, das der Wirklichkeit widerspricht. Statt die Breite der informatischen Anwendungsbereiche – von ‚A‘ wie Archäologie über ‚L‘ wie Lebenswissenschaften zu ‚Z‘ wie Zukunftsfragen – zu betonen, versperrt ein in der Berufswirklichkeit der meisten Informatiker nachrangiger Aspekt den Blick auf einen der vielfältigsten Berufe, der Kommunikations- und Teamfähigkeit, analytisches Denken, Anwendungsorientierung sowie Methoden- und Werkzeugbeherrschung erfordert und in fast allen Bereichen der Gesellschaft präsent ist. Nicht umsonst klagen die universitären Fachbereiche nicht nur über zu wenig Nachwuchs, sondern auch über den falschen. Wir sollten vielleicht besser am Bild der Informatik arbeiten als an immer neuen fachlichen Spezialitäten – und das können wir am besten, wenn all die genannten Aspekte unseren Unterricht prägen.

Um nicht falsch verstanden zu werden: Die Implementierung von eigenen Lösungsideen, das schrittweise Verbessern, auch der experimentelle Zugang zur Problemlösung, der daraus erwachsende Produktstolz und die Förderung selbstständigen Arbeitens sind zentrale Elemente eines lebendigen Informatikunterrichts. Wenn all dieses ohne Syntaxprobleme möglich ist – umso besser!

3 BYOB: Build Your Own Blocks

Der Erfolg von Scratch im Ausbildungssystem ist eigentlich verblüffend: ein für Kindergarten und Grundschule konzipiertes und designtes System wird nicht nur in den Sekundarstufen, sondern sogar in der universitären Grundausbildung erfolgreich eingesetzt – und das nicht in irgendwelchen Klitschen, sondern am MIT oder in Berkeley. Die Not muss also groß sein! Ein „Kindergartensystem“ sollte eigentlich für die universitäre Ausbildung ungeeignet sein. Wird es trotzdem benutzt, dann hält ein Teil der Ausbilder die traditionellen Ausbildungswerkzeuge offensichtlich für noch ungeeigneter.

Die Einschränkungen von Scratch sind in den Bereichen Modularisierung und Datenstrukturen offensichtlich, obwohl lokale und globale Operationen und Variable sowie Zeichenketten und lineare Listen zur Verfügung stehen. Das wurde zum Anlass genommen, ein für die Grundausbildung an Universitäten geeignetes Werkzeug auf der Grundlage von Scratch zu entwickeln. Die Ziele sind im Artikel „Bringing ‘No Ceiling’ to Scratch: Can One Language Serve Kids and Computer Scientists?“⁸ von Brian Harvey und Jens Mönig beschrieben. Mit BYOB ist es möglich, die informatischen Konzepte des Lehrbuchklassikers „Struktur und Interpretation von Computerprogrammen“ von Abelson und Sussman⁹ zu realisieren, also auf der Ebene von Berkeley-Scheme zu arbeiten. Damit wird die konzeptionelle Ebene von Sprachen wie Java deutlich überschritten. Für uns ist aber etwas anderes wichtig: wenn Berkeley seine Informatikvorlesung CS10 im Rahmen des AP-Curriculums¹⁰ mit BYOB als einziger Programmiersprache durchführen kann, dann wird das System auch für die deutsche Sekundarstufe II geeignet sein. Eine Diskussion in dieser Hinsicht erübrigt sich also.

Zentrales Werkzeug von BYOB ist ein Blockeditor, mit dessen Hilfe Konzepte wie strukturierte Zerlegung, geschachtelten Operationen, lokale Variable, Rekursion usw. anschaulich umgesetzt werden. Bei einem Block kann es sich einen neuen Befehl, eine Funktion oder ein Prädikat handeln. Parameter können an beliebiger Stelle eingefügt und bei Bedarf typisiert werden. Mehrere Blockeditoren können gleichzeitig offen sein. Im Bereich der Objektorientierten Programmierung bietet BYOB einen sehr viel verständlicheren Weg als z. B. Java: man kommt von den Objekten zu Klassen, vom Konkreten zum Abstrakten – und nicht umgekehrt. Da BYOB Lisp-Strukturen enthält, ist z. B. die Erzeugung neuer Kontrollstrukturen sehr einfach. Dass die gängigen deutschen Abiturthemen problemlos mit BYOB behandelbar sind, wurde an anderer Stelle gezeigt¹¹.

⁸ [HM10]

⁹ [AS01]

¹⁰ [AP10]

¹¹ [MSM11]

BYOB stellt mit diesen und darüber hinausgehenden Features eine echte Alternative zu textbasierten Entwicklungssystemen dar und ist keineswegs auf den Anfangsunterricht beschränkt. Wir können mit diesem Werkzeug in einem sehr weiten Bereich implementieren, ohne uns um textuelle Syntaxprobleme zu kümmern. Da diese einen Großteil der Unterrichtszeit beanspruchen, hätten wir also sehr viel mehr Zeit als bisher – z. B. zum Implementieren! Wir können unterrichtsbegleitend immer dann schnell lauffähige Programme erzeugen, wenn algorithmisches Problemlösen gefragt ist. Das Werkzeug würde aus dem Fokus verschwinden – so, wie es sein soll.

Wir wollen hier nicht behaupten, dass man deshalb jetzt das Werkzeug wechseln muss, aber wir denken schon, dass man ernsthaft darüber nachdenken kann. BYOB ist also ein Anlass, darüber zu reflektieren, was und weshalb wir etwas in diesem Bereich tun – und das ist doch schön!

4 Was ist nun mit der Syntax?

Natürlich sind Systeme wie BYOB derzeit noch Modellsysteme, anhand derer die Konzepte und Methoden der Informatik übersichtlich und inhaltsorientiert vermittelt und erprobt werden können, und zwar im gesamten Bereich der Schulinformatik. Dafür sind sie gemacht. Nicht gedacht sind sie als echte Produktionssysteme. Für Anwendungsprobleme benötigt man wohl auf absehbare Zeit Entwicklungssysteme wie Eclipse oder Visual Studio mit den entsprechenden textuellen Sprachen, und für einige Gebiete sind diese Werkzeuge natürlich auch einfach besser geeignet.

Wir sind der Meinung, dass die Informatikdidaktik zu begründen hätte, weshalb sich alle Informatikschülerinnen und –schüler mit den Syntaxproblemen textbasierter Programmierung herumschlagen müssen, wenn es dazu Alternativen gibt, die den Schwerpunkt der Arbeit drastisch von der Codierung der Lösung hin zur Entwicklung und Verbesserung von Ideen verlagern. Es wäre zu zeigen, wo der allgemeinbildende Wert textueller Codierung liegt, wenn die Vorteile dieses Verfahrens für die Masse der Lernenden niemals zum Tragen kommen, und es wäre zu zeigen, dass dieser Wert die dadurch zumindest mitverursachte unglaubliche Ausstiegsquote der fast immer freiwillig zu uns Kommenden rechtfertigt.

Andererseits hat die Schule neben der Vermittlung inhaltlicher Kompetenzen auch eine Orientierungsfunktion. Die Lernenden sollten im Schulsystem die Gelegenheit bekommen, sich auf unterschiedlichen Gebieten zu erproben und begründete Perspektiven für ihren Lebensweg zu entwickeln, z. B. bezüglich ihrer beruflichen Orientierung¹². Wenn die textbasierte Programmierung in der fachlichen Informatikausbildung auch eine wesentliche Rolle spielt, dann sollten diejenigen Schülerinnen und Schüler, die sich für diesen Weg entscheiden, wissen, worauf sie sich einlassen. Sie sollten erproben können, ob ihnen diese Tätigkeit liegt. Dieser Aspekt ähnelt dem Vorgehen im Physikunterricht der Oberstufe: die Physik wird weitgehend ohne große mathematische Vorkenntnisse vermittelt, aber die Lernenden sollten an geeigneten Stellen erfahren, dass Physiker später Mathematik brauchen, sie beherrschen und ihre Anwendung auch mögen müssen.

¹² [Mo05a]

Weiterhin geben die informatischen Entwicklungswerkzeuge einigen Schülerinnen und Schülern die Möglichkeit, sich unabhängig von der beruflichen Entwicklung auf diesem Gebiet zu entfalten. Auch diese brauchen ihre Chance!

Wann und wie also kommen wir zu den textuellen Sprachen? Ein sequentielles Vorgehen, also erst BYOB, dann z. B. Java, erscheint uns ungeeignet. Gegenüber der Mächtigkeit und den Anwendungsmöglichkeiten von BYOB kann eine nach anspruchsvollen BYOB-Modellierungen eingeführte textbasierte Sprache nur verlieren, weil die Bedeutung der dann wieder herangezogenen Anfängerproblemchen hinter dem Berg von Syntax- und Hantierungsproblemen verschwindet und sie eigentlich nur demonstrieren, dass textbasierte Programmierung dafür völlig unangemessen ist.

Die Benutzung reduzierter Sprachen mit vereinfachter textueller Syntax und/oder vereinfachtem Befehlssatz scheint auch nicht der rechte Weg zu sein, wenn wir zu universellen Produktionssystemen hin wollen, und für den Einsatz von voll entwickelten Sprachen innerhalb virtueller Miniwelten wie bei JavaKara oder Greenfoot gilt das Gleiche, weil sie den Spracherwerb in den Mittelpunkt stellen und nicht die Algorithmen – wenn diese an anderer Stelle wesentlich übersichtlicher implementiert werden können. So schön z. B. Greenfoot auch ist – es wird genauso wenig und aus den gleichen Gründen nicht als „richtige Informatik“ akzeptiert wie BYOB.

Wenn wir textbasierte Sprachen im Unterricht parallel zu visuellen nutzen, dann muss deren Gebrauch aus der Situation heraus den Lernenden sinnvoll erscheinen – und das ist ein hartes Kriterium! Weiterhin müssen die ersten Beispiele so einfach sein, dass die Syntaxprobleme nicht überwiegen. Gehen wir davon aus, dass anfangs nur visuell und am Ende der Ausbildung in nennenswertem Umfang textbasiert programmiert wird, dann muss sich die textuelle Form langsam „einschleichen“, also an sinnvollen Stellen eingeführt werden, an denen ihre Vorteile augenfällig sind – und die gibt es natürlich:

- Wenn es auf das „Look-and-feel“ echter Anwendungen ankommt, also den Entwurf von Oberflächen, die den gewohnten gleichen, dann liefern GUI-Builder wie gehabt einen einfachen, leicht gangbaren Weg, der bei den ersten Arbeitsschritten der visuellen Programmierung gleicht. Der Übergang ist fließend und die Ausstattung der Eventhandler mit Funktionalität kann auf elementarstem Niveau geschehen, etwa bei der Rekonstruktion vorgefundener Elemente.¹³
- Wenn es auf den Umgang mit „richtigen“ Formeln ankommt, also z. B. bei mathematischen Problemen und ggf. ihrer Verknüpfung mit grafischer Visualisierung, die zusätzlich erheblichen Rechenaufwand erfordert, zeigt sich die Leistungsfähigkeit textueller Hochsprachen sofort. Das Gleiche gilt z. B. für durch zahlreiche Fallunterscheidungen umfangreiche Programme, die nicht anspruchsvoller als ihre kürzeren Verwandten, kaum modularisierbar, aber bei grafischer Darstellung irgendwann völlig unübersichtlich sind. Hier zeigen sich dann langsam die Vorteile der textuellen Repräsentation.

¹³ Beispiele dazu z. B. in [Mo05b]

- Wenn es auf die Nutzung externer Expertise etwa durch die Einbindung von Bibliotheken, Zugriff auf externe Ressourcen wie Datenbanken usw. ankommt, benötigt man Systeme, die diese Integration erlauben. Python mit seinen zahlreichen Schnittstellen nach außen ist dafür ein gutes Beispiel.¹⁴

Für dieses Vorgehen gibt es noch einen weiteren guten Grund. Wir haben in der Vergangenheit mehrfach Situationen gehabt, in denen das in den Schulen vorhandene informatische Wissen durch technische Umwälzungen schlagartig entwertet wurde. Ein Grund dafür war, dass es kaum eine didaktische Begründung für das tradierte Vorgehen gab, für das neue natürlich auch nicht, und deshalb die letzten informatischen Entwicklungen sofort als neues „Paradigma“ verkündet wurden. Die fehlende Didaktik verhinderte ein Bewerten und Abwägen der alten gegen die neuen Chancen und einen geordneten Übergang. Die Schulinformatik hat bisher weitaus mehr Kolleginnen und Kollegen durch Frustration verloren als durch Pensionierung! Führen wir also eine für die Unterrichtenden gewohnte textuelle Programmiersprache behutsam an geeigneten Stellen ein, dann bleibt die Expertise auf diesem Gebiet den Schulen erhalten und die neuen Möglichkeiten stehen dazu nicht im Gegensatz, sondern sollten als sinnvolle Ergänzung erscheinen. Wie sich das Verhältnis dann weiter entwickelt, mag die Zukunft entscheiden. In jedem Fall brauchen wir keinen schmerzhaften Bruch.

5 Fazit

Obwohl einiges für einen Übergang zu visuellen Entwicklungsumgebungen spricht, wobei BYOB als ein möglicher Vertreter für diese anzusehen ist, steht diesem der Widerwille vieler am Lernprozess Beteiligter entgegen. Ohne hier eine Lösung zu finden, können wir den Übergang nicht vollziehen, denn es gibt durchaus abschreckende Beispiele für solch ein Vorgehen. Im Physikunterricht werden z. B. Schülerübungsgeräte benutzt, die man nirgends sonst als in der Schule findet. Diese Abkoppelung von der Erfahrungswelt der Unterrichteten könnte mit ein Grund für die Unbeliebtheit des Faches sein. BYOB sollte nicht in eine „Schülerübungsgerät-Rolle“ gedrängt werden.

Wenn es stimmt, dass das Bild von Informatikern durch das Codieren geprägt wird, dann scheint uns hier der Schlüssel zu liegen. Der zumindest anfängliche Verzicht auf das Codieren schafft Zeit im Unterricht – sehr viel Zeit. Nutzen wir diesen Freiraum doch bitte nicht, um noch mehr „Stoff“ in das Fach hinein zu quetschen. Nutzen wir die Zeit lieber für eine an die Lebenswelt angebundene Informatik, für eine Art „Informatik im Kontext“. Die Erfahrungen in der Mittelstufe mit einer im Fächerverbund unterrichteten Informatik, hier: Astronomie und Informatik, die an anderer Stelle geschildert wurden¹⁵, legen es nahe, dass so ein breites Interesse erreicht werden kann. Nutzen wir die Zeit für projektartiges Arbeiten, für Erfahrungen in Teamarbeit, für die Modellierung von „interessanten“ Bereichen – und damit für vertieften Einblick in diese –, die so einer Algorithmisierung zugänglich gemacht werden. Nutzen wir die vorhandenen Werkzeuge klug, mit allen ihren Einschränkungen, um ihren angemessenen Einsatz zu planen.

¹⁴ siehe dazu z. B. [OI10]

¹⁵ [Mo10]

Und zu dieser Angemessenheit der Werkzeuge sollte es zuerst an einigen Stellen, dann zunehmend gehören, dass textbasierte Systeme dort benutzt werden, wo dieses offensichtlich sinnvoll ist – und sonst eben nicht. Wenn dadurch das Bild des Informatikers vielfältiger und interessanter – eben realistischer – wird, wenn textuelle Programmierung als mächtig und für Spezialisten sinnvoll, aber nicht als alles überschattend erfahren wird, dann hätten wir wirklich etwas erreicht.

Literaturverzeichnis

Alle Internetquellen wurden zuletzt am 9.1.2011 geprüft.

- [AP10] AP Principles: <http://www.csprinciples.org/CSPrinciples0310.pdf>
- [AS01] Abelson, H.; Sussman, G.: Struktur und Interpretation von Computerprogrammen, Springer 2001
- [Be08] Becker, H.: USBomat, 2008, <http://hbecker.sytes.net/usbomat/>
- [CM99] Carnegie Mellon University, Alice 1999, <http://www.alice.org/>
- [Fi10] Fischertechnik RoboPro Software, <http://www.fischertechnik.de/desktopdefault.aspx/tabid-39/>
- [HM10] Harvey, B.; Moenig, J.: Bringing “No Ceiling” to Scratch: Can One Language Serve Kids and Computer Scientists?, Constructionism 2010, Paris
- [Le83] Leeds, B.: Echte Programmierer meiden Pascal., DATAMATION 1983, <http://www.bytecruncher.de/witze/compprg2.html>
- [HuU00] Humbert, L.: Bericht zur Lehrerausbildung Informatik, Königstein 2000 <http://koenigstein.inf.tu-dresden.de/00/humbert2.html>
- [MH10] Moenig, J.; Harvey, B.: BYOB 3.0 – Build Your Own Blocks, August 2010, <http://byob.berkeley.edu/>
- [MI07] MIT MediaLab, Scratch 2007, <http://scratch.mit.edu/>
- [Mo05a] Modrow, E.: Informatikunterricht mit technischen Aspekten, MNU 58/7 2005
- [Mo05b] Modrow, E.: Einführung in die Algorithmik, 2005, <http://vlin.de/index.php?p=sek>
- [Mo10] Modrow, E.: Informatik als technisches Fach, LOG IN 163/164 2010
- [MSM11] Modrow, E.; Strecker, K.; Mönig, J.: Wozu Java?, LOG IN 168, 2011 (im Druck)
- [NHR99] Nievergelt, J.; Hartmann, W.; Reichert, R.: Kara, 1999, <http://www.swisseduc.ch/informatik/karatojava/kara/>
- [Ol10] Oldenburg, R.: Programmieren mit Python in der Sek.1, 2010, <http://www.hrpi.gi-ev.de/fileadmin/gliederungen/fg-hrpi/texte/python.pdf>

Agiler Informatikunterricht: Soziale Aspekte der professionellen Softwareentwicklung im Schulunterricht erfolgreich erfahrbar machen

Timo Göttel

Angewandte und Sozialorientierte Informatik
Universität Hamburg
tgoettel@acm.org

Abstract: In diesem Artikel wird diskutiert, warum das Image der Informatik nach wie vor negativ behaftet ist. Es wird die These aufgestellt, dass dies zu Teilen darauf zurückzuführen ist, dass Informatikunterricht nicht in ausreichendem Maße die sozialen Aspekte der heutigen professionellen Softwareentwicklung (SE) berücksichtigt. Das Papier vermittelt, dass bereits beim ersten Kontakt mit Informatik darauf Wert gelegt werden sollte, dass es sich bei der Softwareentwicklung um eine gemeinschaftliche Arbeit handelt, bei der Lösungen immer im Dialog mit anderen Menschen entwickelt werden. Es wird dargelegt, dass in aktuellen Bildungsempfehlungen soziale Aspekte der Informatik zwar genannt werden, jedoch davon auszugehen ist, dass diese mangels Methoden kaum in Schulen vermittelt werden. Diese Ansicht wird gestützt durch Ergebnisse einer Lehrerbefragung. Dieses Papier stellt darüber hinaus dem Schulkontext angepasste Methoden der SE, bzw. der agilen Methoden vor. In Projekten mit Schülerinnen und Schülern wurden diese bereits mehrfach erprobt, so dass in diesem Artikel Hinweise zur erfolgreichen Anwendung gegeben werden können.

1 Einleitung

Die Informatik hat einen schlechten Ruf bei Studienanfängern. Die Anfängerzahlen bleiben deutschlandweit unter der möglichen Aufnahmekapazität der Universitäten. Dies ist besonders erstaunlich, handelt es sich doch bei den heutigen Studienanfängern um *digital natives* [Pr01], die mit der Informationstechnologie (IT) aufgewachsen sind und diese ausgiebig nutzen. Trotzdem sehen Jugendliche allem Anschein nach in der Informatik noch immer das befremdliche Berufsfeld, das nur für Solisten attraktiv ist [Ga09]. Demgegenüber sind gerade soziale Netzwerke beliebte und selbstverständlich genutzte Anwendungen von Jugendlichen. Offenbar als natürlich erscheint es ihnen, dass IT die soziale Interaktion, wie sie sie kennen und lieben, ermöglicht und gestaltet. Für sie ergibt sich daher auch nicht, dass das IT-Berufsbild einen ähnlichen interaktiven dynamischen und zwischenmenschlichen Charakter hat, der sich z.B. auch in der Art und Weise von social web Anwendungen widerspiegelt.

Die Informatik besitzt viele soziale und kreative Aspekte [Ro08b], die meist dem gemeinschaftlichen Lösen von Problemen dienen. Davon findet sich noch recht wenig im Schulunterricht wieder. Während es z.B. nach Romeike vielversprechende Ideen, Methoden und

Werkzeuge gibt, um Kreativität passend zu vermitteln [Ro08a], findet man in der Literatur kaum Hinweise auf Methoden oder Werkzeuge, um die sozialen Aspekte abzudecken. Es ist davon auszugehen, dass dieses Versäumnis auch einen Anteil an dem eingangs beschriebenen schlechten Ansehen der Informatik hat: Überspitzt formuliert, sitzen im Unterricht alle Schülerinnen und Schüler vor ihren Computern, und es gibt wahrscheinlich zwei bis drei Personen, die schnell Lösungen auf irgendwie magische Weise und aus dem Nichts erarbeiten. Das Bild der Einzeldisziplin, bei der kryptische Anweisungen in den Computer gehackt werden, ist geboren. Dieses Bild steht im Gegensatz zu Berufsbildern, die man in der professionellen Softwareentwicklung (SE) findet, wo Lösungen immer häufiger in Teamarbeit und durch eine Vielfalt von sozialer Interaktion entstehen. Naheliegender erscheint es daher zu untersuchen, wie die sozialen Aspekte der heutigen SE im Unterricht erfahrbar gemacht werden können. Besonders eignen sich hier als Bezugspunkt die agilen Methoden, die besonderen Wert auf soziale Interaktion und dynamische Prozesse legen. Diese Grundprinzipien sind in dem agilen Manifesto prägnant formuliert [Be01].

Im Weiteren wird zur Veranschaulichung der Problematik zunächst die Diskrepanz aufgezeigt, die zwischen der schulischen Informatik und den aktuellen Strömungen im IT-Berufsfeld herrscht. Es werden mögliche Gründe anhand von Bildungsstandards und einer Lehrerbefragung ermittelt. Lösungsansätze zur Überwindung der Diskrepanz werden daraufhin erarbeitet, indem Anleihen bei der SE gemacht und deren Methoden in einen schulischen Kontext gebracht werden. Die aus Projekterfahrungen entstandenen Tipps zur erfolgreichen Anwendung der Methoden werden zusätzlich vorgestellt.

Festzustellen bleibt jedoch, dass sich das vorherrschend negative Image der Informatik nicht nur durch die hier aufgezeigten Methoden allein bekämpfen lässt. Man muss sich darüber im Klaren sein, dass sehr viele Aspekte und gesellschaftliche Prozesse beteiligt sind. Die genannten Methoden sollen daher nur als ein Schritt in die richtige Richtung verstanden werden, in der Hoffnung, dass weitere Ansätze darauf aufbauend in der Praxis entwickelt werden.

2 Informatik an der Schule

Bei den folgenden Betrachtungen sei natürlich immer darauf verwiesen, dass sich diese an den verfügbaren Empfehlungen und Bildungsstandards orientieren. Engagierte Lehrer vermögen es bereits ohne die hier vorgeschlagenen Methoden, die formulierten Inhalte und sozialen Aspekte anschaulich zu vermitteln. Zahlreiche Beobachtungen und Gespräche sowohl mit Schülerinnen und Schülern als auch mit Lehrern im Rahmen von Projektveranstaltungen ergaben jedoch, dass es gerade in der Vermittlung von sozialen Aspekten und besonders der Gruppenerfahrungen der Informatik an Methoden und Ideen zu mangeln scheint. Dieser Eindruck wurde zudem durch eine Umfrage unter Informatiklehrern im Rahmen eines Workshops gefestigt (siehe dazu Kapitel 2.1).

Im Folgenden soll auf Publikationen zum Informatikunterricht an Schulen eingegangen werden. Besonderes Augenmerk wurde hierbei auf soziale Aspekte gelegt. In diesem Zusammenhang ist besonders darauf zu achten, ob und wie die geforderten sozialen Aspekte auch der SE entsprechend widerspiegelt werden können. Prinzipiell sei gesagt, dass

dieser Artikel in der Überzeugung verfasst wurde, dass bereits in Empfehlungen und Bildungsstandards Elemente enthalten sein sollten, die zur Steigerung der Attraktivität des Fachs beitragen. Ein Fokus auf die reine Zweckmäßigkeit und die großen Berufschancen dürfte nicht ausreichend sein, um IT-Berufe in ein besseres und zutreffenderes Licht zu stellen.

In den Grundsätzen und Bildungsstandards für die Informatik, die durch die Gesellschaft für Informatik e.V. (GI) herausgegeben wurden¹, finden sich einige Hinweise darauf, dass Informatik als ein Schulfach begriffen werden sollte, das besondere interdisziplinäre und soziale Aspekte aufweist [Pu08]. Hierbei fällt auf, dass besonders zu Anfang der Publikation darauf Wert gelegt wird, dass Lösungen für informatische Probleme sehr häufig im Team erarbeitet werden müssen. Daher wird verlangt, dass ein guter Informatikunterricht den Schülerinnen und Schülern das Selbstvertrauen dazu vermittelt. Ein weiterer Aspekt ist, dass es den Verfassern darauf ankommt, dass die informatische Schulausbildung vorbereitet auf *“das Leben, so wie es ist”*. Die Autoren siedeln ihre Empfehlungen bewusst zwischen Input- und Output-Orientierung an, trotzdem sind gerade Hinweise zu passenden Methoden zur Vermittlung von sozialen Aspekten schwer auffindbar. Es ist verwunderlich, dass immer wieder darauf hingewiesen wird, dass künftige Lebenssituationen (a.a.O., S.1, S.4, S.6) berücksichtigt werden sollen, aber nie darauf verwiesen wird, dass es durchaus Möglichkeiten gibt, sich der Elemente der SE zu bedienen, um dies realitätsnah zu gestalten. Auch wird darauf eingegangen, dass gemeinsames Erarbeiten von informatischem Wissen gemeinsam stattfinden soll. Begründet wird dies jedoch meist damit, dass so möglichst effektiv Informatikstoff vermittelt werden kann; auch hier finden sich keine Anhaltspunkte, dass der Bezug zur realen SE die Attraktivität des Fachs steigern könnte. Der Artikel stellt heraus, dass nur in der Informatik die Möglichkeit für Schülerinnen und Schüler besteht, zu erfahren, *“ob die konstruktive Arbeit mit technischen Werkzeugen für sie möglich und attraktiv, eben eine Lebensperspektive ist”* und somit auch die reale SE an Schulen erfahrbar sein muss. Die Interdisziplinarität der Informatik wird darüber hinaus erwähnt, jedoch nicht als Faktor der Informatik erkannt, der die Attraktivität steigern könnte; vielmehr wird nur darauf verwiesen, dass so fächerübergreifende Stoffe vermittelt werden können. Hier wäre eine offensivere Herangehensweise wünschenswert, die es vermag, gerade den nicht von Haus aus an der Informatik interessierten Schülerinnen und Schülern das Fach schmackhaft zu machen. Eine genauere Untersuchung nach sozialen Aspekten lohnt sich bei den folgenden identifizierten Prozessbereichen: “Modellieren und Implementieren”, “Kommunizieren und Kooperieren” und “Darstellen und Interpretieren”. Auffallend bei “Modellieren und Implementieren” ist, dass ein deutlicher Fokus auf Werkzeuge gelegt wird und dabei nicht erwähnt wird, dass gerade diese Arbeit nur noch sehr selten als Einzelleistung in der SE zu verstehen ist. Hier scheint die Möglichkeit vertan, den Schülerinnen und Schülern zu vermitteln, dass gerade das Erarbeiten dieser Lösungen einen hochgradig kreativen und zwischenmenschlichen Aspekt der IT-Berufsbilder darstellt. Gerade die Forderungen bei “Kommunizieren und Kooperieren” erscheinen als sehr treffend. Betrachtet man jedoch die Praxis, so scheint es einen Mangel an Methoden zu geben, die Lehrern an die Hand gegeben werden können (siehe hierzu nochmals 2.1). In den Betrachtungen zu “Darstellen und Interpretieren” werden leider nur selten Hinwei-

¹ Die GI spricht hierbei von einem Mindestmaß an informatischem Wissen, das spätestens mit der mittleren Reife erlangt werden soll und somit breiten Anklang im zukünftigen Gesellschaftsbild finden soll.

se gegeben, dass gerade diese Fähigkeiten eigentlich nur in Kleingruppen erlernt werden können, um so z.B. zu erfahren, wie Darstellungsformen von anderen Teilnehmern interpretiert werden.

In den Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen, herausgegeben durch die GI, wird darüber hinaus auch die Vermittlung und Hervorhebung von Sozialkompetenz in den Aufgabenbereich der Informatikbildung gerückt. Es wird hierbei gezielt gefordert, dass Schülerinnen und Schüler befähigt werden sollen, *“Gruppenprozesse zu planen und mitzugestalten, Kritik entgegenzunehmen bzw. konstruktiv formulieren zu können, einen Arbeitsrollenwechsel zu erleben und akzeptieren zu können”* [Br00]. In diesem Text findet sich sogar der konkrete Hinweis, dass diese Fähigkeiten auch in der realen (IT-) Arbeitswelt wiederzufinden sind. Es wird jedoch nicht darauf verwiesen, dass diese Gegebenheit selbst auch ein zu vermittelndes Wissen ist.

In der Empfehlung der Kultusministerkonferenz (KMK) zur Stärkung der mathematisch-naturwissenschaftlich-technischen Bildung finden sich manche Forderungen, die in ihrer Interpretation und Durchführung zum Ansehen der Informatik und auch der Qualitätssicherung im Informatikunterricht beitragen können [Ku09]. Darin wird darauf verwiesen, dass möglichst *“konkrete Erfahrungen mit naturwissenschaftlichen, ingenieurwissenschaftlichen und technischen Berufen zu ermöglichen”* sind. Hierbei wird zwar in erster Linie darauf Wert gelegt, dass Netzwerke zu Firmen entstehen sollen, so dass sich Jugendliche vor Ort davon ein Bild machen können. Im Umkehrschluss ist jedoch auch denkbar, dass eben konkrete Methoden aus dem Berufsbild in den Schulkontext überführt werden sollten. Weiteren Anreiz bietet auch das in der Empfehlung der KMK beschriebene Handlungsfeld *“Gesellschaftliche Akzeptanz”*, in dem die Rede davon ist, dass das Interesse an technischen Fragestellungen bei Schülerinnen und Schülern zu wecken sei. Hier ist es natürlich verwunderlich, dass nicht auch darauf hingewiesen wird, welche kreativen, sozialen und interdisziplinären Berufsbilder in diesem Bereich vorhanden sind.

Betrachtet man die oben genannten Empfehlungen und Hinweise, so liegt es nahe zu behaupten, dass es mangels konkreter Methoden wohl sehr schwierig für Lehrer ist, diese Inhalte angemessen zu vermitteln und sie somit nicht ihren Weg in die Schulpraxis finden. Um diesen Eindruck zu überprüfen, wird hier eine Lehrerbefragung herangezogen.

2.1 Lehrerbefragung

In einem Fragebogen wurden, abzielend auf die oben beschriebenen sozialen Aspekte im Informatikunterricht, 17 Informatiklehrer aus Hamburg und Schleswig-Holstein befragt. 13 Teilnehmer waren Gymnasial-, zwei Real- und einer Gesamtschullehrer, vier davon waren weiblich. Die Studie wurde von uns im Anschluss an einen Workshop durchgeführt, die Teilnahme war freiwillig.

Auf die Frage, in welchen Veranstaltungsformen Informatikinhalt durch die befragte Person vermittelt wurden, ergab sich folgende Verteilung (Mehrfachnennungen waren erlaubt): drei als Pflichtfach, zwölf als Wahlpflicht, zwei Wahlfach, sechs AG und nur eine Projektwoche. Diese Antworten sind bemerkenswert, da die Mehrzahl in Wahlpflicht und Wahlfach liegt. Naheliegender ist, dass zu diesem Zeitpunkt bereits eine Vorauswahl der Schülerinnen und Schülern stattgefunden hat. So ist davon auszugehen, dass nur noch eine gewisse, ohnehin schon informatikaffine Klientel teilnimmt. Wünschenswert wäre eine

Vielzahl von Projektangeboten, da dort meistens ein breiteres Spektrum an Schülerinnen und Schüler beteiligt ist und somit auch die Chance besteht, andere Zielgruppen für die Informatik zu begeistern. Darüber hinaus dürften gerade Projektwochen und AGs besonders gut den Projektcharakter der SE vermitteln dürfen.

Bei der Frage, welche Inhalte vorwiegend in den Lerneinheiten programmiert wurden (Mehrfachnennungen waren auch hier erlaubt), gaben sieben Teilnehmer an, dass Anwendungen entwickelt wurden, sechs bezeichneten Algorithmen als Hauptfokus und niemand identifizierte Games als Lerninhalt. Darüber hinaus gab es noch andere Anwendungen die Verwendung fanden und extra genannt wurden: PovRay, Robotik / Lego Mindstorms, Robot Karol und Microsoft Office. Positiv bleibt festzuhalten, dass es einen Fokus auf Anwendungen gab (die ja auch Algorithmen beinhalten können). So ist wenigstens davon auszugehen, dass prinzipiell auf Fragen der SE Bezug genommen werden kann. Unter dem Aspekt der Kreativität in der Informatik ist es jedoch enttäuschend, dass niemand die gute Gelegenheit nutzt und mit Games das aufgreift, was die meisten Jugendliche begeistern dürfte und darüber hinaus einen guten Anknüpfungspunkt an aktuelle Entwicklungen in der SE darstellt.

In der Befragung zu verwendeten Web-Plattformen zur Unterstützung der Arbeit der Schülerinnen und Schüler, ergab sich folgendes Bild: sechs boten das Schul-CommSy an, sechs lo-net², fünf gaben andere, nicht weiter spezifizierte an. Erfreulich ist hier ganz klar, dass dementsprechend alle Teilnehmer solche webbasierte Unterstützungen anbieten, die natürlich (wenn auch wohl in leicht anderer Form) Einsatz in der SE finden.

Aus der offenen Frage, ob Gruppenarbeit stattfindet und wenn ja, wie diese gestaltet sei, ergibt sich, dass Gruppenarbeit von nahezu allen Lehrern umgesetzt wurde (im Schnitt mit einer Gruppenstärke von zwei bis drei Teilnehmern), es jedoch häufig an Methoden mangelt, um diese zur Zufriedenheit der Lehrer und der Schülerinnen und Schüler durchzuführen. Es wurde geäußert, dass es gerade an der Aufgabenverteilung unter den Teilnehmern hapere, da es schwerfällt, dort das richtige Herangehen zu vermitteln. Darüber hinaus entsteht aus den Antworten der Eindruck, dass manche Gruppenarbeit trotzdem bedeutet, dass jeder Teilnehmer Aufgaben an einem eigenen PC bearbeitet. Dies zeigt insgesamt ganz klar, dass hier ein Verbesserungsbedarf besteht, der die sinnvolle Gruppenarbeit in geordnete Bahnen bringt. Auch hier liegt es nahe, sich der Projektmanagementmethoden der SE zu bedienen und dies auch unter diesem Aspekt den Schülerinnen und Schülern zu vermitteln.

Bei den Antworten zu der Frage, ob Dokumentationen angefertigt werden müssen und zu welchem Zeitpunkt dies geschehe, ist herauszulesen, dass diese eher zum Ende hin angefertigt werden müssen und diese Aufgabe von sehr wenigen Schülerinnen und Schülern geschätzt wird. Somit wird deutlich, dass attraktivere Herangehensweisen vorgestellt werden müssen, um Programmiererergebnisse im schulischen Kontext bewerten zu können. Hier gibt die SE keine offensichtliche Antwort, da es dort nicht um Einzelleistungen, sondern um ein fertiges und funktionierendes Produkt geht. Trotzdem soll versucht werden, Artefakte aus der SE, die parallel zur Programmierung entstehen, zumindest in Teilen zur Dokumentation und Präsentation zu verwenden (siehe Kapitel 4.3).

Abschließend kann man zusammenfassen, dass es gerade für die oben beschriebenen sozialen Aspekte und den Realitätsbezug an einfachen Methoden zu mangeln scheint, die Lehrern mit auf den Weg gegeben werden können, um Gruppenarbeit zu vereinfachen und strukturiert verlaufen zu lassen. Darüber hinaus scheint es jedoch wichtig, dass der spannende und kreative Entwicklungsprozess im Vordergrund steht und störende Nebenaufgaben, wie z.B. Abschlussdokumentationen interessanter gestaltet werden bzw. entsprechend der SE während des eigentlichen Projekts nebenläufig entstehen. Wie bereits erwähnt, bieten agile Methoden Techniken an, um diese Probleme zu bewältigen.

3 Agile Methoden in der professionellen Softwareentwicklung

Agile Methoden folgen einem Manifest, das 2001 von Kent Beck et al. verfasst wurde und bereits vorhandene populäre Herangehensweisen und Denkweisen der SE zusammenfassen sollte, indem es ein besonderes Augenmerk auf Werte wie z.B. Kommunikation und Feedback innerhalb der Gruppe legt [Be01]. Es entstand aus dem Wissen, dass große Projekte vormals immer darunter litten, dass Auftraggeber und Entwickler schnell unvereinbare Sichten auf das Produkt entwickeln, wenn die SE nach klassischen Modellen, wie z.B. dem Wasserfallmodell aufgebaut ist. Ein Projektmanagement, das dem Manifest folgt, war demzufolge gefragt. Den Autoren zufolge sind Softwareentwicklungsprozesse durch agile Methoden mehr auf sozialen Austausch fokussiert, und die entstehenden Produkte sind hierdurch besser an die Bedürfnisse der Nutzer angepasst. Wobei zu sagen ist, dass es den Autoren neben sozialeren Arbeitsstrukturen auch bzw. wohl in erster Linie um eine effektivere Beziehung zwischen Auftraggeber und Entwickler geht.

Bis zum heutigen Tage haben sich zwei Varianten der Agilen Methoden besonders etabliert: Extreme Programming (XP) nach Beck [Be00] und Scrum, das aus mehreren Arbeiten heraus entstanden ist, wobei wohl drei Publikationen den größten Anteil daran zu haben scheinen [TN86, Be99, SB01]. Beide Varianten versuchen möglichst allumfassend, den gesamten Prozess der SE abzudecken und passende Methoden anzubieten. Die folgenden Beschreibungen von XP und Scrum sind lediglich auf soziale Strukturen bzw. Methoden fokussiert, um den ursprünglichen Punkt dieses Papiers nicht aus den Augen zu verlieren.

4 Agile Methoden an Schulen

Programmiertechnische Vorgehensweisen der agilen Methoden wurden in Schulkontexten bereits erfolgreich eingesetzt [We05]. Im Gegensatz dazu betrachtet der vorliegende Artikel die sozialen Aspekte der Arbeitsprozesse der agilen Methoden. Im Folgenden werden entsprechende Methoden aus XP und Scrum vorgestellt, die bereits in mehreren Projekten an Schulen und ähnlichen Einrichtungen erfolgreich verwendet wurden. Es handelt sich um Projekte, bei denen die grafische Programmierumgebung Scratch² oder Greenfoot³ verwendet wurde, um interkulturelle Geschichten am Computer zu erzählen [Gö09a]. For-

² <http://scratch.mit.edu/>, zuletzt besucht am 14.04.2011

³ <http://www.greenfoot.org/>, zuletzt besucht am 14.04.2011

schungsziel der Projekte war es, aus dem Verhalten der Schülerinnen und Schüler Werkzeuge zu entwickeln, um die dynamischen und zwischenmenschlichen Aspekte der Programmierung erfahrbar zu machen und zu unterstützen. Erste daraus entstandene prototypische Werkzeuge liegen als Ergänzung von Scratch um kollaborative Funktionalitäten vor und sind Gegenstand eines anderen Artikels [Gö09b].

Die Methoden werden nachfolgend aus der Sicht der SE beschrieben und dann gegebenenfalls mit Hinweisen versehen, wie die jeweilige Methode in einem schulischen Kontext anzuwenden ist, und welche Probleme auftreten können und wie diesem entgegengewirkt werden kann.

Es sei empfohlen, diese Methoden in einer Auftaktveranstaltung vorzustellen und den klaren Bezug zu den realen agilen Methoden herzustellen. Besonders geeignet zur Steigerung der Attraktivität der Methoden scheint der Verweis auf die Grundwerte der agilen Methoden, die auch in einem schulischen Kontext von großer Bedeutung sein dürften: Kommunikation, Respekt, Mut, Einfachheit und Feedback [Be00].

4.1 Pair programming

Die bekannteste Methode von XP dürfte das *pair programming* sein. Die Idee dabei ist, dass jeweils vor einem Rechner zwei Entwickler gemeinsam programmieren. In der Praxis bedeutet dies, dass eine Person die Tastatur verwendet und somit den Quellcode schreibt, wobei die zweite Person die Arbeit hinterfragt, auf Fehler prüft und alternative Lösungsvorschläge macht. Es ist wichtig, dass die Person mit Tastatur die eigenen Gedanken bei der Arbeit mündlich mitteilt, um der zweiten Person das Erkennen von möglichen Denk- und Strukturfehlern zu erleichtern. Idealerweise beginnt die Person mit Tastatur erst zu schreiben, wenn sich das Paar einig über die Lösung ist. In der Regel werden die Tastatur und damit die Rollen mehrmals untereinander im Laufe des Arbeitstages getauscht.

Im schulischen Kontext ist darauf zu achten, dass die Rolle der Person ohne Tastatur ernsthaft eingenommen wird und dass es zu dem Wechsel der Tastatur wirklich kommt. Hier ist eine zeitliche Taktung sehr hilfreich. Darüber hinaus ist bei Paaren mit stark unterschiedlichem Wissenstand darauf zu achten, dass die Person mit einem geringeren Kenntnisstand die Tastatur häufiger verwendet und der Partner sich darin übt, sein Wissen zu vermitteln. Hierbei ist besonders acht zu geben, dass nicht einfach die Anweisungen diktiert werden, sondern dass Konzepte erklärt werden. Notfalls hilft es hier Paare zu tauschen.

In mehreren Projekten konnte beobachtet werden, dass gerade Schülerinnen das Programmieren im Paar sehr schnell annehmen und häufig davon begeistert sind, dass gemeinsam mit einem Partner Lösungen erarbeitet werden. Dies ist wohl einer von vielen Schritten, um weibliche Jugendliche für IT-Berufe zu begeistern. Studien mit weiblichen Studierenden lassen zumindest die Hoffnung zu, dass solche sozialen Aspekte der SE das Selbstbewusstsein bezüglich der eigenen Informatikfähigkeiten stärken, das benötigt wird, um sich einen IT-Beruf zuzutrauen [Be04].

4.2 User Stories

Es handelt sich bei *user stories* um die kleinteilige und prägnante Beschreibung von elementaren Funktionalitäten, die einem Endnutzer mit dem fertigen Softwareprodukt zur Verfügung stehen sollen. Zu finden sind *user stories* sowohl bei XP als auch bei Scrum.

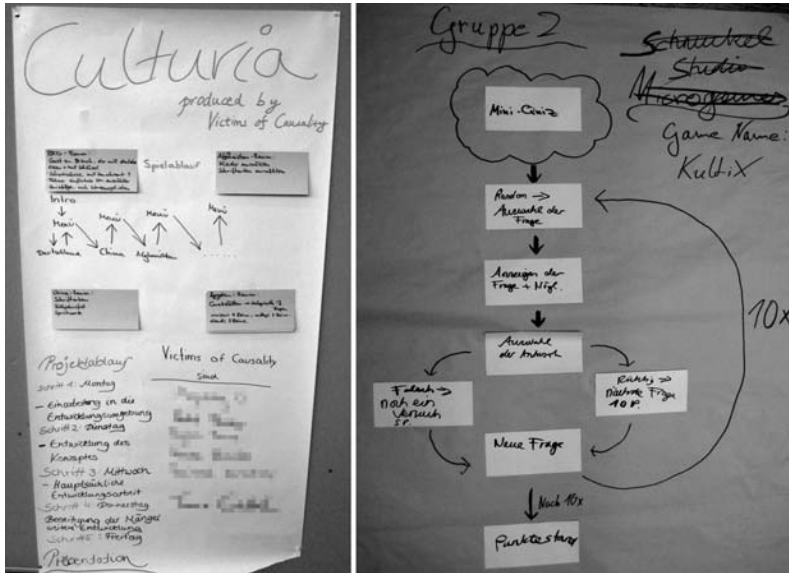


Abbildung 1: Die Poster des *informative workspace* können als Diskussionsgrundlage, zur Präsentation und zur Dokumentation verwendet werden (Namen nachträglich anonymisiert)

Ziel ist es, diese elementaren Anforderungen nicht aus dem Auge zu verlieren, wobei die *user stories* gleichzeitig zur Organisation der Aufgaben innerhalb des Entwicklungsteams dienen. Häufig werden einzelne *user stories* dann von Programmierpaaren übernommen und fertiggestellt. Erst nach Fertigstellung widmet sich ein Paar neuen *user stories*.

Im schulischen Kontext handelt es sich wohl um eine Methode, die gerade zu Anfang eine Hilfestellung benötigt, da Schülerinnen und Schüler häufig dazu tendieren, möglichst alle Funktionalitäten in einer Beschreibung abdecken zu wollen. Elementare Aufgaben erscheinen ihnen schnell als zu banal. Daher empfiehlt es sich für den Anfang, Beispiele vorzugeben und gemeinsam mit den Teilnehmern zu erarbeiten, dass diese kleinteilige Herangehensweise die Gruppenarbeit erleichtert, da nur so Aufgaben verteilt werden können und niemand befürchten muss, dass immer nur ein anderer Teilnehmer die spannenden und fordernden Aufgaben vermittelt bekommt, wohingegen man selbst entweder zu banale oder aber unlösbare Aufgaben zugeteilt bekommt.

4.3 Informative workspace

Bei XP wird empfohlen, dass sämtliche Prozesse und Lösungswege auf Postern skizziert werden. Diese Poster sollen in einem allgemein zugänglichen Raum zur Verfügung stehen, so dass sich Teilnehmer jederzeit über den aktuellen Stand der Entwicklung informieren können und darüber hinaus eine gemeinsame Diskussionsgrundlagen haben. Den meisten Schülerinnen und Schülern ist auf Anhieb nicht ersichtlich, wozu sie solche Poster anfertigen sollen. Es empfiehlt sich daher, ihnen klar zu machen, dass solche Poster die Arbeitswege darstellen und somit sehr gut verwendet werden können für mögliche Präsentationen und Abschlussdokumentationen (siehe beispielsweise Abb. 1). Es ist empfehlenswert, für

die Veranstaltungsleiter von Zeit zu Zeit auf die Poster einzugehen, gezielt Fragen zu stellen, ob diese noch aktuell sind, und Diskussionen anhand der Poster zu führen.

4.4 Meetings

Eine Methode, die man bei XP und in Scrum findet, ist ein tägliches Treffen zu Beginn des Arbeitstages (XP: *stand up meeting*, Scrum: *daily scrum meeting*), bei dem alle Projektmitglieder kurz der Gruppe mitteilen sollen, was am vergangenen Tag geschafft wurde, wo Probleme lagen und welche Entwicklungen für den aktuellen Tag vorgesehen sind.

Hier muss gerade zu Beginn stark darauf geachtet werden, dass jede Person individuelle Erfahrungen mitteilt. In der Klassensituation entsteht häufig eine Dynamik, in der auf den Vorredner verwiesen wird. Dies ist möglichst zu unterbinden, in dem man die Wichtigkeit deutlich macht, dass die Gruppe einen Eindruck erlangen soll, dass jeder für das gemeinsame Ziel arbeitet.

5 Zusammenfassung und Ausblick

Die hier genannten Techniken der agilen Methoden sollen es durch ihren klaren Berufs- und Realitätsbezug erlauben, Schülerinnen und Schülern zu verdeutlichen, welche Chancen IT-Berufsbilder darstellen. Darüber hinaus sollen diese Methoden aber auch Gruppenarbeit im Informatikunterricht unterstützen. Gerade im Bereich der Gruppenunterstützung mangelte es bislang an leicht anzuwendenden Methoden, die sich für Lehrer, Schülerinnen und Schüler als attraktiv darstellen und auch einfach anzuwenden sind. Sicherlich sind diese Methoden nicht in allen Unterrichtssituationen einsetzbar, da dort häufig auch andere Kriterien von Belang sind. Wünschenswert wäre es, dass während der Unterrichtsvorbereitung immer wieder hinterfragt wird, ob der Einsatz der genannten Methoden möglich ist. In der Regel ergibt sich daraus ein Mehrwert durch die dynamisch gestaltete Gruppenarbeit bei nur sehr geringen Einsatzhürden.

Anzuraten ist, dass diese Methoden mit dem klaren Bezug zur SE vorgestellt werden und zu Anfang aktiv unterstützt werden. Es bleibt zu sagen, dass die Einführung dieser Methoden immer auch mit Widerwillen (gerade unter den männlichen fortgeschritteneren Teilnehmern) verbunden sein können; hier ist es natürlich angeraten, die Vorteile der Methoden weiterhin glaubhaft und überzeugend zu vermitteln. Ganz im Sinne des agilen Manifestos (“interactions over processes and tools”) ist es jedoch wichtig, dass diese Methoden bei Bedarf gemeinsam mit den Teilnehmern angepasst werden können. Mischformen, die sich an speziellen Bedürfnissen orientieren erscheinen sinnvoll. Sie vermögen es, dynamische Prozesse, wie sie in der SE stattfinden, abzubilden und sind so ein weiterer Faktor, um dem Berufsbild Informatik zu einem passenderen Image zu verhelfen.

Es kann nur angeraten werden, noch weitere Methoden zu adaptieren und wenn möglich immer ein Ohr an den aktuellen Entwicklungen der SE zu haben. Nur so erscheint es möglich, die starke Diskrepanz zwischen Realität und Ansehen der Informatik wenigstens in Teilen abzubauen. Darüber hinaus müssen weitere Schritte unternommen werden, will man langfristig das gesellschaftliche Ansehen der Informatik verbessern. Ein Ansatz ist, gezielt auf die vielen verschiedenen Aspekte der Informatik einzugehen (siehe

dazu [BCM06]) und diese früh und altersgerecht an Schulen zu vermitteln. Gerade die Chancen, die durch die enorme Interdisziplinarität der Informatik entstehen, scheinen nach wie vor nicht bei Schülerinnen und Schülern angekommen zu sein.

Die Methoden haben in den bisher durchgeführten Projekten mit Schülerinnen und Schülern dazu beigetragen, dass die Teilnehmer mit Spaß an klar geregelten Aufgaben in der Gruppe arbeiten konnten. Auf diesem Weg konnten sie für sich feststellen, dass es sich bei der Informatik um eine soziale interaktive Disziplin handelt.

Literaturverzeichnis

- [Be01] Beck, K. et al.: <http://agilemanifesto.org/> (zuletzt besucht am 14.04.2011), 2001.
- [BCM06] Biundo, S.; Claus, V.; Mayr, H. C., Hrsg: *Was ist Informatik? Unser Positionspapier*. Gesellschaft für Informatik e.V., 2006.
- [Be99] Beedle, M. et al.: *Scrum: A Pattern Language for Hyperproductive Software Development*, Seiten 637–651. Addison Wesley, 1999.
- [Be00] Beck, K.: *Extreme programming explained: embrace change*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.
- [Br00] Breier, N. et al.: *Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen*. Gesellschaft für Informatik e.V. (GI), 2000.
- [Be04] Berenson, S. B. et al.: Voices of women in a software engineering course: reflections on collaboration. *Journal on Educational Resources in Computing*, 4(1):3, 2004.
- [Gö09a] Göttel, T.: *Cultura: Intercultural Learning by Designing Games*. The Inter-Disciplinary Press, 2009.
- [Gö09b] Göttel, T.: Virtual sandbox: adding groupware abilities to Scratch. In *Proceedings of the 8th International Conference on Interaction Design and Children*, Seiten 158–161, Como, Italy, 2009. ACM New York, NY, USA.
- [Ga09] García-Crespo, Á. et al.: IT Professionals' Competences: High School Students' Views. *Journal of Information Technology Education*, 8:45–57, 2009.
- [Ku09] Kultusministerkonferenz (KMK), Hrsg. *Empfehlung der Kultusministerkonferenz zur Stärkung der mathematisch-naturwissenschaftlich-technischen Bildung*. 2009.
- [Pu08] Puhlmann, H. et al.: *Grundsätze und Standards für die Informatik in der Schule*. Gesellschaft für Informatik e.V. (GI), 2008.
- [Pr01] Prensky, M.: Digital Natives, Digital Immigrants. *On the Horizon*, 9(5), 2001.
- [Ro08a] Romeike, R.: Towards students' motivation and interest: teaching tips for applying creativity. In *Proceedings of the 8th International Conference on Computing Education Research*, Seiten 113–114, Koli, Finland, 2008. ACM.
- [Ro08b] Romeike, R.: Sichtweisen einer Kreativen Informatik. In Torsten Brinda, Michael Fothe, Peter Hubwieser und Kirsten Schlüter, Hrsg., *Didaktik der Informatik - Aktuelle Forschungsergebnisse*, Seiten 129–138. Gesellschaft für Informatik e.V. (GI), 2008.
- [SB01] Schwaber, K.; Beedle, M.: *Agile Software Development with Scrum*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st. Auflage, 2001.
- [TN86] Takeuchi, H.; Nonaka, I.: New New Product Development Game. *Harvard Business Review*, 1986.
- [We05] Weigend, M.: Extreme Programming im Klassenraum, INFOS 2005 Dresden. Erweiterte Version. http://www.fernuni-hagen.de/schulinformatik/xp_weigend.pdf (zuletzt besucht am 14.04.2011), 2005

Das Schülerlabor als Ort der Informatiklehrerbildung

Carsten Schulte

Didaktik der Informatik
Freie Universität Berlin
Königin-Luise Str. 24-26
14195 Berlin
schulte@inf.fu-berlin.de

Abstract: Im Artikel wird anhand der internationalen Diskussion über Lehrerbildung und mithilfe des Ansatzes der Phänomenographie ein Konzept zur Integration von Schülerlaboren in die Informatiklehramtsbildung vorgestellt, um so die Praxisbezüge der ersten Phase der Lehrerbildung zu stärken. Gleichzeitig werden Grundlagen gelegt, um durch einen ‚forschenden Blick‘ auf Unterricht Kompetenzen zur Planung, Durchführung und Reflexion besser als ohne solche Praxisbezüge zu entwickeln.

1. Einleitung

„Das Problem der Lehramtsausbildung hier an der Uni ist, dass es so wenig Schüler gibt“, so beschrieb ein Student einmal, was seiner Meinung nach in der ersten Phase der Lehrerbildung fehlt. Damit hat er einen Kern getroffen, den man als das Theorie-Praxis-Problem bezeichnen kann, welches in unterschiedlichen Varianten auftritt: Das Lehramtsstudium sei zu wissenschaftsorientiert und bereite schlecht auf die berufliche Praxis vor (Problem der Berufsbezogenheit). Fachdidaktische Innovationen kommen nur schwer im Schulalltag an, unter anderem weil konkrete Materialien und Hilfen zur Umsetzung fehlen (Problem der Dissemination). Fachdidaktische Konzepte stellen eine Art „Feiertagsdidaktik“ dar, die im Schulalltag kaum weiterhelfe (Problem der Relevanz).

Im Folgenden wird ein Konzept vorgestellt, welches diese Probleme des theorieorientierten fachdidaktischen Studiums durch die Integration von Praxiserfahrungen zu überwinden bzw. zumindest abzumildern versucht. Dabei geht es nicht um eine Ausweitung schulpraktischer Anteile, sondern um Überlegungen, wie in der universitären Lehre Praxiserfahrungen konzipiert und integriert werden sollen, damit Lehramtsstudierende Theorie und Praxis verknüpfen können. Die Idee ist, Schülerlabore so in das Lehramtsstudium zu integrieren, dass diese als Lern- und Lehrlabore das Theorie-Praxis-Problem überbrücken helfen. Dazu werden theoriegeleitet unterrichtsbezogene Prozesse initiiert, praktisch von den Studierenden erfahren und anschließend ausgewertet. Gegenüber Praxissemestern und schulpraktischen Erfahrungen stehen hier also nicht authentische Erfahrungen im Berufsfeld im Mittelpunkt, sondern das Verbinden des theoretisch erworbenen fachdidaktischen Wissens mit konkreten fachlichen Lernprozessen von Schülerinnen und Schülern.

2. Ausgangslage

Lehrerbildung ist ein komplexes und unübersichtliches Feld. Beispielsweise listet die wissenschaftliche Suchmaschine Google Scholar im Januar 2011 etwa 19.500 Publikationen zum Suchbegriff „Teacher Education“ – und das sind nur die Publikationen des vorangegangenen Jahres. Einen Überblick geben: [NJS06], [CEV07], [CC01], [OE09], [OD08] und [CP07].

Nach [Ko10] ist das Theorie-Praxis-Problem bereits seit Dewey 1904 ein zentrales Problem der Lehrerbildung. Die Anwendung von Theorie in der Praxis sei jedoch aus mehreren Gründen problematisch: Erstens ist Praxis komplex, daher ist es nicht einfach, einen jeweils ‚passenden‘ theoretischen Ansatz zu finden. Zweitens wird die theorieorientierte Erstausbildung durch die Sozialisation in der Berufspraxis überlagert und teilweise verdrängt. Drittens spielen das Vorwissen aus der eigenen Erfahrung als Schülerin oder Schüler oft eine bestimmende Rolle, und viertens erschwere der Zeitdruck in der unterrichtlichen Situation das Anwenden abstrakter Theorien. Anhand dieser bzw. aus ähnlichen Problembeschreibungen werden ganz unterschiedliche Konzepte abgeleitet:

- Korthagen [Ko10] schlägt vor, Lehramtsstudierende in Eins-zu-Eins Situationen (d.h. ein Student unterrichtet einen Schüler) unterrichten zu lassen, um so eine Brücke von der Theorie zur Praxis zu schlagen.
- Sykes u.a. [SBK10] schlagen in einem von ihnen als „englische Lösung“ bezeichneten Vorschlag vor, die Lehramtsausbildung aus der Universität zu lösen und vorrangig an der Schule anzusiedeln, um so auch auf spezifische kommunale Bedürfnisse eingehen zu können.
- Ure [Ur10] schlägt ein Konzept vor, in dem durch Praxiserfahrungen in kleinen Gruppen unter Anleitung und Supervision durch erfahrene Lehrkräfte und Experten aus der Lehrerbildung eine stärkere Theorie-Praxis-Verzahnung erreicht werden soll. Ein wichtiger Baustein des Konzepts ist die Auswertung, in der auf wertendes („richtendes“) Feedback verzichtet werden soll.
- Nach Niemi und Jakku-Sihvonen [NJS06] hat sich als Stärke der finnischen Lehrerbildung die starke Theorieorientierung erwiesen, die sich durch wissenschaftlichen bzw. einen forschenden Blick auf Unterrichtsprozesse auszeichnet. So sollen Praxiserfahrungen im Umfang von 20 ECTS erworben werden, mit dem Ziel, „Fähigkeiten zur Erforschung, Entwicklung und Evaluation von Lehr- und Lernprozessen“ (aaO, S.57) zu entwickeln. Ebenso gehören eine Einführung in empirische Forschungsmethoden und die Durchführung eines eigenen didaktischen Forschungsprojekts zum Pflichtkanon der Lehrerbildung.
- Nach dem Motto „Teach as you preach“ schlagen Struyven u.a. [SDJ10] vor, die didaktisch-methodische Ausgestaltung der universitären Lehrveranstaltungen in Didaktik an denjenigen Prinzipien auszurichten, die den Lehramtsstudenten als wesentlich für das spätere eigene Unterrichten vermittelt werden, also etwa beispielsweise offenes, problembasiertes Lernen, Arbeit in Kleingruppen usw. in der universitären Lehre einzusetzen. Die Ergebnisse sind jedoch gemischt - d.h. dass nicht alle Studierenden davon profitieren bzw. dass der Lernerfolg von individuellen Voraussetzungen abhängt.

Die bis hier genannten Ansätze aus der Diskussion um Lehrerbildung können natürlich nur ausschnitthaft sein. Sie zeigen verschiedene Ansätze und Ergebnisse, wie das Theorie-Praxis-Problem angegangen werden könnte – allerdings nur recht schematisch bzw. kaum selbst theoretisch bzw. konzeptuell in eine Lerntheorie der Lehrerbildung eingeordnet. Eine solche Einordnung soll im nächsten Schritt mittels des Ansatzes der Phänomenographie erfolgen.

3. Phänomenographie als lerntheoretische Grundlage

Die im vorangegangenen Abschnitt referierten Ergebnisse und Vorschläge verweisen auf folgendes Problem: Unterrichten ist mehr als das Anwenden von Theorien, Konzepten oder Rezepten. Unterrichten wird durch die eigene Persönlichkeit bestimmt, ist Handeln und Agieren in einer kommunikativen Situation mit anderen Menschen und bezieht daher auch den Unterrichtenden als Person insgesamt ein. Mit anderen Worten: die individuellen Werthaltungen, Überzeugungen, Einstellungen und Weltbilder, die ‚beliefs‘ und ‚attitudes‘, die subjektiven Theorien über Lehren und Lernen spielen eine wesentliche Rolle, ob und wie welche theoretischen Wissens Elemente als praxisrelevant gesehen werden.

Die individuelle Vorstellung bzw. Wahrnehmung vom Unterrichten bestimmt also, wie ein Lehrer als Unterrichtender agiert. In der Phänomenographie (abgekürzt als P) werden eben solche Wahrnehmungen untersucht. Also nicht der Unterricht (abgekürzt als U) selbst, sondern die Wahrnehmung des Unterrichts ($L \rightarrow U$) durch den Lehrer (L).

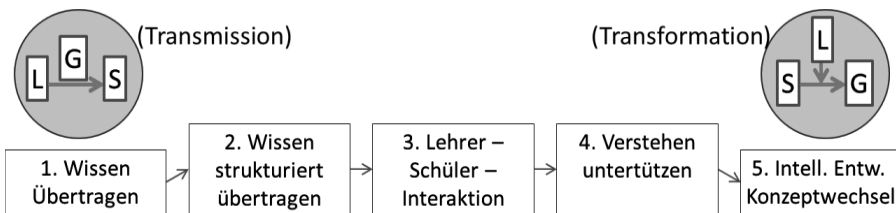


Abbildung 1 Vorstellungen über Lehren, nach [Ke97]¹

Der phänomenographische Ansatz und die darin gewonnenen Ergebnisse der lehramtsbezogenen Lehr-Lernforschung und Lehrerinnenforschung bilden die theoretische Grundlage für das Konzept. Phänomenographie entstand in den 1980er Jahren in Schweden. Ein zentrales frühes Ergebnis ist die Unterscheidung von ‚oberflächlichem Lernen‘ und ‚tiefem/vertiefendem Lernen‘ (surface learning vs. deep learning). Dieses Ergebnis zeigt bereits den typischen Charakter phänomenographischer Forschung. In dieser wird untersucht, wie unterschiedlich Menschen ein Phänomen verstehen können. Das Ergebnis zeigt dann qualitativ unterschiedliche Konzeptualisierungen eines Gegenstands (hier: Lernen). In der P. wurde ebenfalls Unterrichten als Gegenstand untersucht.

¹ Allerdings beziehen sich diese Ergebnisse auf Lehrer an akademischen Institutionen. Lehrer an Sekundarschulen haben [BL+01] untersucht, sie kommen zu sehr ähnlichen Ergebnissen, wobei sie zwischen den beiden Polen 4 verschiedene Abstufungen unterscheiden konnten.

Ergebnis ist die Unterscheidung von Lehren oder Unterrichten als Informationsvermittlung (Transmission) einerseits, oder als Sinnkonstitution (Transformation) andererseits (vgl. Abbildung 1).

Transmission entspricht dem ‚Nürnberger Trichter‘, während der andere Pol mit konstruktivistischen oder sozio-konstruktivistischen Vorstellungen übereinstimmt. Siehe dazu Abbildung 1: Linker Kreis, Transmission: Der Lehrer (L) ‚transportiert‘ den zu lernenden Gegenstand (G) - in geeigneter Form - zu den Schülerinnen und Schülern (S). In der zweiten Variante (Transformation, rechter Kreis) setzen sich die S direkt mit G auseinander, wobei diese Begegnung durch L gestaltet bzw. beeinflusst wird. Insgesamt beschreibt [Ke07] neben diesen beiden Polen fünf verschiedene Konzeptualisierungen des Unterrichtens. Diese werden durch ‚kritische Aspekte‘ genauer erläutert. In Kember’s Meta-Studie über das Phänomen Unterrichten sind das: Lehrer, Lehren, Lerner, Lernen, Gegenstand und Wissen. Jeder dieser kritischen Aspekte kann unterschiedlich wahrgenommen werden – der Lernende als aktiv oder als ‚passiver Empfänger‘. Die einzelnen Aspekte stehen dabei in einem Zusammenhang: Beispielsweise passt die Sichtweise, Lehren ist Informationsvermittlung gut zur Sichtweise, dass Lernende eher passive Empfänger sind, und ebenso zur Vorstellung, dass Wissen vom Lehrer zum Schüler übertragen wird, der Lehrer also (alleine) im Besitz des Wissens ist. Auf diese Weise wird die Dimension ‚Transmission‘ durch die wesentlichen, in der Sprechweise der P die ‚kritischen‘ Aspekte genauer beschrieben. Für die detaillierten Ergebnisse sei auf [Ke07] verwiesen.

Für die Lehrerbildung interessant ist, dass diese verschiedenen Konzeptualisierungen mit weiteren Variablen verknüpft sind: Trigwell u.a. [TPW99] zeigen in einer empirischen Studie einen Zusammenhang zwischen „beliefs“ über Lehren und Lernen und der Art, wie Lernende lehren. Demnach korrespondieren oberflächliche Lernstrategien der Lernenden mit einer der Lehrvorstellung der Transmission des Unterrichtenden; ebenso korrespondieren tiefe Lernstrategien (deep learning) eher mit lerner- und lernprozessorientierten Lehrvorstellungen.

Van Driel u.a. [DBV07] zeigen in einer empirischen Studie mit Chemielehrerinnen und -lehrern aus den Niederlanden, dass Vorstellungen über Unterricht auch mit Einschätzungen bzw. Vorstellungen über das Curriculum korrespondieren. Demnach schätzen ‚transmissionsorientierte‘ Lehrende eher ein Curriculum, das den Schwerpunkt auf Vermittlung der Kernkonzepte des Fachs legt. Lehrer mit Vorstellungen aus dem gegenüberliegenden Pol schätzen eher ein Curriculum, das kontextorientierte und gesellschaftliche Implikationen des Fachs einbezieht.

Die Ergebnisse dieser Studien können m.E. gut die ‚gemischten‘ Ergebnisse der Studie von [SDJ10] erklären, wonach ‚teach as you preach‘ manchmal Wirkung zeigt und manchmal nicht: Das Vermitteln von Wissen über Lehr- und Lernprozesse und Unterrichtsmethoden gelingt nur, wenn die Weltbilder der Studierenden ‚passen‘ – diese müssen also in der universitären Lehramtsausbildung stärker als bisher allgemein üblich berücksichtigt werden. Da in der P. die Wahrnehmung von Phänomenen im Mittelpunkt steht, stellt sie Ansätze bereit, den Wechsel in diesen Wahrnehmungen lerntheoretisch zu fassen. Daher kann sie die Grundlage bilden, ein Konzept zu entwickeln, das neben der Vermittlung von Theorie und Praxiserfahrung das Überwinden der Lücke zwischen

Theorie und Praxis versucht, indem den Lernprozess beeinflussende Weltbilder bzw. Wahrnehmungen der Lehramtsstudierenden in das Lehrkonzept einbezogen werden.

Lernen bedeutet in der P. einen qualitativen Wechsel in der Konzeptualisierung – einen Wechsel im Weltbild. Man kann sich hier zunächst also etwa den Wechsel von einer Dimension in die nächste vorstellen (vgl. die Pfeile in Abbildung 1). Allerdings zeichnet sich ein Experte nicht nur durch einen ‚Konzeptwechsel‘ von Dimension A nach B aus, sondern durch ein größeres Repertoire an unterschiedlichen Konzeptualisierungen und ein besseres Vermögen, unterschiedliche kritische Aspekte je nach Bedarf in den Fokus der Wahrnehmung zu bringen – und dazu kann ein Experte auch bewusst zwischen den Dimensionen wechseln (um die jeweils passende „konzeptuelle Brille“ zu verwenden). Mit anderen Worten: Aus Sicht der P kann die informationsvermittelnde Sichtweise manchmal durchaus die angemessene Perspektive sein, um über Unterricht und Lernen nachzudenken.

Wie man die unterschiedlichen Lernvoraussetzungen und Wahrnehmungen über den Lerngegenstand (hier: Unterrichten) berücksichtigen kann wurde in neueren Weiterentwicklungen der P untersucht. Dazu sei hier auf die Stichworte ‚phenomenographic pedagogy‘ ([PT97], [TPG05]) und ‚variation theory‘ [Ru06]) verwiesen – einige konkrete Ausführungen folgen weiter unten, wenn drei spezifische Lehrveranstaltungsmodulare beschrieben werden. Allgemein lässt sich sagen, dass es beim Lernen darum geht, die Wahrnehmung zu schärfen, etwa in dem ein bestimmter kritischer Aspekt bewusst erfahren wird. Als didaktische Mittel werden Kontrastierung, Generalisierung, Separation und Fusion vorgeschlagen [Th10].

4. Konzept

Das Konzept sieht vor, Schülerlabore als Lehr-Lernlabore in die Lehramtsausbildung zu integrieren. Schülerlabore sind an der Hochschule betriebene Einrichtungen, in denen Schülerinnen und Schüler, insbesondere im Klassenverband, interessante und aktuelle ‚Schnupperkurse‘ in z.B. Informatik belegen können. Diese Angebote dauern zumeist nur wenige Stunden und werden teilweise mit Lehrerfortbildungen und/oder Materialien zur schulischen Vor- bzw. Nachbereitung ergänzt.

Schülerlabore ermöglichen also, an der Universität mit Schülerinnen zu agieren und so Lehr- und Lernprozesse zu erleben und zu erfahren. Schülerlabore sollen insgesamt drei Mal mit unterschiedlichen Schwerpunkten in universitäre Lehrveranstaltungen integriert werden. Hier sind natürlich diverse unterschiedliche Einbettungen denkbar.

Die folgende Abbildung beschreibt das Konzept aus phänomenographischer Perspektive: Lernen ist demnach die Auseinandersetzung eines Schülers (S) mit dem Gegenstand (G), der gelernt werden soll. Die Begegnung des Schülers mit dem Gegenstand wird durch den Lehrer (L) beeinflusst, der dazu mit dem Schüler interagiert, und sich natürlich ebenso mit dem Gegenstand des Unterrichts auseinandersetzt. Aus phänomenographischer Perspektive sind nun insbesondere die beschriebenen Relationen interessant, die in der Grafik mit Zahlen gekennzeichnet sind.

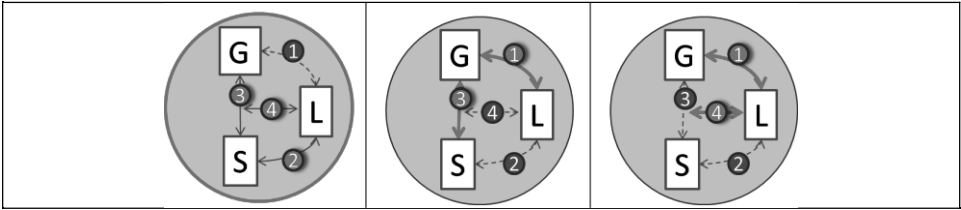


Abbildung 2: Überblick, die Kreise stellen die Schwerpunkte der drei Module dar

Im ersten Schritt (in der Grafik der erste Kreis links) sollen Lehramtsstudierende ihre Wahrnehmung des Phänomens Unterricht schärfen und insbesondere über ②, ③ und ④ reflektieren. Im ersten Schritt ist der fachliche Gegenstand noch im Hintergrund, während in den folgenden beiden Schritten ① – die fachdidaktische Durchdringung des Lerngegenstands – jeweils wesentlich ist. Im zweiten Schritt steht zudem ③ im Mittelpunkt: das Wahrnehmen und Erkennen unterschiedlicher Arten, wie Schüler (in der jeweiligen Unterrichtsumgebung) lernen. Im dritten Schritt steht nicht das Lernen sondern das Lehren ④ im Mittelpunkt, die verschiedenen theoriegeleiteten Möglichkeiten, die Begegnung von S mit dem Lerngegenstand G zu gestalten und zu steuern. Im folgenden Abschnitt werden die drei Lehrveranstaltungen genauer charakterisiert.

5. Drei Module

Intention in allen Modulen ist, einen ‚forschenden Blick‘ auf Unterrichtsplanung und -durchführung zu gewinnen. Es soll darum gehen, die wichtigen Aspekte wahrzunehmen, deren Variation sowie Muster in der Variation – d.h. den Zusammenhang der Aspekte. Die Lehramtsstudenten sollen verstehen, dass es nicht darum gehen kann, Rezepte zu lernen, wie informatische Inhalte vermittelt werden können. Alle drei Module wurden erprobt (jedoch nicht in einer formalen empirischen Studie evaluiert), sodass in die folgende Beschreibung Erfahrungen aus der Durchführung eingeflossen sind.

5.1. Modul 1 - Informatikunterricht planen, durchführen und reflektieren

Im ersten Modul geht es darum, die wesentlichen Aspekte des Phänomens Unterricht ins Wahrnehmungsfeld zu bringen. Das sind – siehe Abbildung 2, linke Seite – die Beziehung zwischen Schüler und Lerngegenstand ③, die aufgrund der Beziehung von Lehrer und Schüler ② durch den Lehrer gestaltet wird ④.

Der Lerngegenstand, zu verwendende Werkzeuge sowie methodische Grundsätze, Ziele und ein theoretischer Rahmen werden relativ eng vorgegeben. Studierende entwickeln gemeinsam nach diesen Vorgaben Informatikunterricht, erproben ihn und werten ihn aus. Der Versuch soll von den Studierenden als Erprobung eines Unterrichtskonzepts verstanden werden. Nicht die eigene Erfahrung als Lehrperson ist zentrales offizielles Veranstaltungsziel (obschon Kompetenzgewinne intendiert sind), sondern der forschende Blick, indem eine zuvor entwickelte Unterrichtsumgebung erprobt wird. Dementsprechend sollen sich Feedback und Auswertung auf die Umgebung und weniger auf die

Lehrperson beziehen. Leitfragen dienen als Fokussierung der Auswertung und als Beobachungskriterien für die Studierenden, die gerade nicht selbst unterrichten. Sie beziehen sich auf die Dimensionen ②, ③ und ④. Die Auseinandersetzung mit dem Gegenstand ① bleibt im Hintergrund, wird jedoch nicht ausgeblendet. Sie wird vor allem in der Vorbereitung berücksichtigt, indem das zu verwendende Unterrichtsmaterial (zumindest teilweise) selbst entwickelt beziehungsweise auch programmiert wird – diese Entwicklungstätigkeiten unterstützen und motivieren die detaillierte Sachanalyse.

Während der Durchführung werden die gehaltenden Stunden jeweils anhand der Leitfragen reflektiert und die Planung angepasst. Hier sind Debatten zwischen Lehrperson und betreuendem Dozenten hilfreich, um auf diese Weise die Variation der Dimensionen zu erfahren, und damit Merkmale als zugehörig identifizieren zu können.

Hier einige studentische Kommentare zur Veranstaltung: 1) „ist mir klar geworden, dass die Lehrkraft nicht die Aufgabe hat, den Schülerinnen und Schülern [G] zu erklären, sondern, dass die Schüler aktiviert werden um sich selber Gedanken darüber zu machen und sich gegenseitig ergänzend [G] erklären.“ 2) Der Student fand interessant, welche ‚merkwürdigen‘ Begründungen und Argumente über mögliche Unterrichtsmethoden es gibt und dass diese zumeist doch irgendwie sinnvoll erscheinen. Sehr spannend fand er, die unterschiedlichen Sichtweisen der Fachdidaktiker, Fachwissenschaftler und Fachlehrer konkret in Diskussionen erleben zu können (Variation, Dimension 4). Spannend war die Konsequenz, dass man sich getraut habe, Unterrichtsmethoden auszuprobieren, die man sonst nie verwendet hätte. 3) Ein Student stellte von sich aus Dimension 2 in den Fokus und protokollierte deren Variation und erstellte ein Schaubild, das in der Gruppe diskutiert wurde. Phänomenographisch ausgewertet, konnten also tatsächlich verschiedene kritische Aspekte des Unterrichts wahrgenommen, nämlich ②, ③ und ④, sowie deren Variation erfahren werden.

5.2. Modul 2 - Lernprozesse im Kontext beobachten

In diesem Modul liegt der Schwerpunkt auf der Planung und Beobachtung von fachlichen Lernprozessen der Schülerinnen und Schüler ③. Themenschwerpunkt ist die Algorithmik, die nach dem IniK-Ansatz vermittelt wird. Beispielsweise indem ein Algorithmus aus der Bioinformatik in diesem Kontext vermittelt wird. Das Modul besteht aus einer Planungsphase, der Durchführungsphase, die evaluiert und wiederholt wird, und einer abschließenden Reflexionsphase. In der Vorbereitungsphase sollen zunächst die eigenen Lernprozesse der Lehramtsstudierenden bewusst gemacht werden. Dazu müssen diese sich in einen relativ komplexen Algorithmus (durchaus etwas über Schulniveau) einarbeiten ① und diesen für eine Laborveranstaltung aufbereiten ④ (Schwerpunkt auf Planung). Im Labor werden verschiedene Schülergruppen (ggf. parallele Lerngruppen) diesen Algorithmus erarbeiten.

Die Vorbereitungsphase besteht im Einzelnen aus der Sachanalyse, der didaktisch-methodischen Analyse inkl. des Entwurfs der Laborveranstaltung sowie der Konstruktion von Beobachtungsaufgaben und von Hypothesen über mögliche Lernverläufe und Lernergebnisse. Die Planungsphase konzentriert sich zunächst auf die Analyse des Gegenstands ①, indem verschiedene Algorithmen mittels Blockanalyse einer Sachanalyse

unterzogen werden. Für den ausgewählten Algorithmus wird diese um eine didaktisch-methodische Analyse ergänzt, in der ausgehend von der Blockanalyse das Unterrichtskonzept entworfen wird ④ (darbietendes oder erarbeitendes Vorgehen, Reihenfolge der einzelnen Lernschritte, Entwicklung von Aufgaben und insbesondere das Bestimmen der Lernziele (vgl. [BS10])). Insbesondere soll für den konkreten Algorithmus geklärt werden, was mit „Die Schülerinnen und Schüler verstehen den Algorithmus“ gemeint ist. Das leitet in den dritten Schritt der Planungsphase, in der Hypothesen über mögliche Lernverläufe und Lernergebnisse sowie Beobachtungsaufgaben entwickelt werden.

Während der Durchführung und Auswertung sollen die am Seminar teilnehmenden Lehramtsstudierenden hauptsächlich beobachten. Wenn möglich, unterrichten deshalb ggf. andere Lehrer (In unserem Fall sind es Bioinformatikstudierende, die den Unterricht durchführen und dabei von den Lehramtsstudierenden beobachtet werden). Schwerpunkt ist das Beobachten und Nachvollziehen der Lernprozesse von Teilgruppen oder ggf. einzelnen Schülern. In der dazugehörigen Auswertung stehen die verschiedenen Lernprozesse und Lernergebnisse im Mittelpunkt. Ziel ist das Erkennen der Variation des Lernens. Diese sollen dann u.a. mittels Schemata wie die oben angesprochenen Lernstile (oberflächliches, tiefes Lernen ...) einsortiert und erklärt werden ③.

In der Reflexionsphase werden die beobachteten Lernprozesse gesammelt und auf den konkreten Inhalt bezogen kontrastierend gegenübergestellt. Ziel ist, so auch Erkenntnisse über Lernhürden, Lernsequenzen und ertragreiche theoriebasierte Erklärungsmuster zu sammeln. Hier werden ③, ④ und ① in Beziehung gesetzt. Wesentlicher Punkt ist dabei die Einarbeitung der gewonnenen Erkenntnisse über die Lernprozesse der Schülerinnen und Schüler. Dieses Vorgehen ist mit fachdidaktischer Forschung und Entwicklung verknüpft, da aus dem Blockmodell [BS10] sukzessive, durch die Erfahrungen im Seminar „Sequenzschablonen“ abgeleitet werden; sowie Materialien für verschiedene Algorithmen entstehen (z.B.: Sequenzvorschläge, Blockanalysen, Aufgaben, ggf. Berichte mit Beschreibungen der Lernprozesse).

5.3. Modul 3 - Unterricht entwickeln: Fachinhalte nach dem Stand der Fachdidaktik aufbereiten

In diesem Modul sollen verschiedene curriculare Aufbereitungsmöglichkeiten eines Inhalts erfahren werden können, und die mit der didaktisch-methodischen Aufbereitung intendierten Lernprozesse in Bezug auf den Prozess und das Produkt des Lernens (The What and the How aspect of learning [MB97]) vor dem Hintergrund fachdidaktischer Ansätze bzw. Theorien erfahren und untersucht werden. Im Mittelpunkt stehen die Beziehungen ① und ④ (Abbildung 2, rechts): Die Sachanalyse des Gegenstands sowie die Planung von Unterricht. In beiden Fällen steht die praktische Anwendung von Theorie (noch stärker als in den beiden vorangegangenen Modulen) im Vordergrund.

Die Stoff- bzw. Themenaufbereitung (Sachanalyse) kann als Prozess der ‚didaktischen Reduktion‘, bzw. ‚Elementarisierung‘ verstanden werden. Diese Sichtweise deckt sich mit dem Transmissionsmodell und einer stoffzentrierten Sichtweise auf den Lehrprozess. Der andere Pol der Transformationsorientierung fragt dagegen eher nach der Sinnkonstitution durch die Lernenden (vgl. [DBV07], Diskussion im Abschnitt 3). Im Modul

wird daher die Art der Unterrichtsplanung variiert. Dazu werden drei verschiedene, aktuelle fachdidaktische Ansätze als Werkzeuge zur curricularen Aufbereitung der Fachinhalte verwendet (Fusion von ① und ④). Dadurch soll fachdidaktische Theorie berufsbezogen und konkret erfahren und mit der Sachanalyse verknüpft werden. Um das genaue Wahrnehmen zu unterstützen, werden andere kritische Aspekte konstant gehalten – eine Veranstaltung des Unterrichtslabors wird also z.B. dreimal nach verschiedenen Theorien entworfen (dieselbe Zielgruppe, dasselbe Thema, derselbe organisatorische Rahmen), und die Ergebnisse werden anschließend gegenübergestellt und bilden den Ausgang für die Gestaltung des tatsächlichen Unterrichtens. Dadurch können die Studierenden verschiedene curriculare Positionen in ihrer Variation erleben.

Die Durchführung legt den Schwerpunkt auf die Beobachtung des Lernprozesses und des Ergebnisses des Lernens. Die Lehramtsstudierenden sollen die Korrespondenzen von fachdidaktischem Modell und Unterrichtsmethodik als zusammenhängend erleben, dazu den Prozess dokumentieren und dabei die zuvor erarbeiteten Beobachungskriterien berücksichtigen. In einem dritten Schritt findet die Überarbeitung als Verbesserung des Ansatzes statt, dient ebenso aber als Argumentationshilfe für theoriegeleitete Ansätze.

6. Fazit

Eine qualitativ hochwertige Lehrerbildung befähigt angehende Informatiklehrerinnen und -lehrer, guten Informatikunterricht zu planen, durchzuführen, zu reflektieren und nicht zuletzt auch dazu, Informatikunterricht zu innovieren. Lehrerbildung muss adäquat fachwissenschaftliche, fachdidaktische und pädagogische Grundlagen legen und diese verknüpfen. Im Artikel wurde ein Konzept zur Integration von Schülerlaboren in die Informatiklehrerbildung vorgestellt, um so die Praxisbezüge der ersten Phase der Lehrerbildung zu stärken. Gleichzeitig sollen Grundlagen gelegt werden, um durch einen ‚forschenden Blick‘ auf Unterricht Kompetenzen zur Planung, Durchführung und Reflexion besser als ohne solche Praxisbezüge zu entwickeln.

Die Arbeit entstand im Rahmen des von der Telekom-Stiftung geförderten FU.MINT-Projekts. (<http://www.fu-berlin.de/mint-lehrerbildung/>)

Literaturverzeichnis

- [Ak08] G. Åkerlind, “A phenomenographic approach to developing academics' understanding of the nature of teaching and learning,” *Teaching in Higher Education*, Jg. 13, Nr. 6, S. 633-644, 2008.
- [BL+01] G. M. Boulton-Lewis, D. J. H. Smith, A. R. McCrindle, P. C. Burnett, and K. J. Campbell, “Secondary teachers' conceptions of teaching and learning,” *Learning and Instruction*, Jg. 11, Nr. 1, S. 35-51, Feb. 2001.
- [BS10] T. Busjahn, C. Schulte: Das Blockmodell als Hilfsmittel zur fachdidaktischen Analyse von Quelltexten. In: Marco Thomas, Michael Weigend (Hrsg.): *Informatik und Kultur*. 4. Münsteraner Workshop zur Schulinformatik. 2010 ZfL-Verlag, S. 11-21
- [CC01] The Carnegie Corporation of New York, “Teachers for a New Era. A national initiative to improve the quality of teaching,” 2001.
- [CEV07] B. M. Standing Committee on Education and Vocational Training, *Top of the Class - Report on the inquiry into teacher education*. House of Representatives Publishing

- Unit, 2007.
- [CP07] Committee on Prospering in the Global Economy of the 21st Century: An Agenda for American Science and Technology, National Academy of Sciences, National Academy of Engineering, Institute of Medicine, *Rising Above the Gathering Storm: Energizing and Employing America for a Brighter Economic Future*. The National Academies Press, 2007.
- [DBV07] J. H. Van Driel, A. M. Bulte, and N. Verloop, "The relationships between teachers' general beliefs about teaching and learning and their domain specific curricular beliefs," *Learning and Instruction*, Jg. 17, Nr. 2, S. 156–171, 2007.
- [EGS10] H. Ezer, I. Gilat, and R. Sagee, "Perception of teacher education and professional identity among novice teachers," *European Journal of Teacher Education*, Jg. 33, Nr. 4, S. 391-404, 2010.
- [Ke97] D. Kember, "A reconceptualisation of the research into university academics' conceptions of teaching," *Learning and instruction*, Jg. 7, Nr. 3, S. 255–275, 1997.
- [Ko10] F. Korthagen, "How teacher education can make a difference," *Journal of Education for Teaching*, Jg. 36, Nr. 4, S. 407-423, 2010.
- [MB97] Ference Marton, Shirley Booth: *Learning and Awareness*, LAWRENCE ERLBAUM ASSOCIATES, PUBLISHERS, 1997
- [NJS06] H. Niemi, R. Jakku-Sihvonen, "In the Front of the Bologna Process Thirty Years of Research-based Teacher Education in Finland," in *Education Science*, 2006.
- [OD08] J. Osborne and J. Dillon, "Science education in Europe: Critical reflections," London: Nuffield Foundation, 2008.
- [OE09] *Creating Effective Teaching and Learning Environments: First Results from TALIS* – ISBN 978-92-64-05605-3, OECD 2009
- [PT97] M. Prosser and K. Trigwell, "Using Phenomenography in the Design of Programs for Teachers in Higher Education," *Higher Education Research & Development*, Jg. 16, Nr. 1, S. 41-54, 1997.
- [Ru06] U. Runesson, "What is it Possible to Learn? On Variation as a Necessary Condition for Learning," *Scandinavian Journal of Educational Research*, Jg. 50, Nr. 4, S. 397-410, 2006.
- [SBK10] G. Sykes, T. Bird, M. Kennedy, "Teacher Education: Its Problems and Some Prospects," *Journal of Teacher Education*, Jg. 61, Nr. 5, S. 464-476, 2010.
- [SDJ10] K. Struyven, F. Dochy, and S. Janssens, "'Teach as you preach': the effects of student-centred versus lecture-based teaching on student teachers' approaches to teaching," *European Journal of Teacher Education*, Jg. 33, Nr. 1, S. 43-64, 2010.
- [Th10] E. Thompson, "From phenomenography study to planning teaching," in *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, S. 13–17, 2010.
- [TPG05] K. Trigwell, M. Prosser, P. Ginns, "Phenomenographic pedagogy and a revised Approaches to teaching inventory," *Higher Education Research and Development*, Jg. 24, Nr. 4, S. 349-360, 2005.
- [TPW99] K. Trigwell, M. Prosser, F. Waterhouse, "Relations between teachers' approaches to teaching and students' approaches to learning," *Higher Education*, Jg. 37, Nr. 1, S. 57-70, Jan. 1999.
- [Ur10] C. L. Ure, "Reforming teacher education through a professionally applied study of teaching," *Journal of Education for Teaching*, Jg. 36, Nr. 4, S. 461-475, 2010.

Medienkunde + Informatik = ?

Zur Neukonzeption des Kurses Medienkunde an Thüringer Schulen

Bernd Bethge, Dirk Drews, Ursula Rumpf

Thüringer Institut für Lehrerfortbildung,
Lehrplanentwicklung und Medien (Thillm)
Heinrich-Heine-Allee 2-4
99438 Bad Berka

{ bernd.bethge, dirk.drews, ursula.rumpf }@thillm.de

Michael Fothe, Gabor Meißner

Friedrich-Schiller-Universität Jena
Fakultät für Mathematik und Informatik
Ernst-Abbe-Platz 2
07743 Jena

{ michael.fothe, gabor.meissner }@uni-jena.de

Abstract: Eine erfolgreiche Implementierung von Medienkunde an Schulen benötigt eine verbindliche schulinterne Planung und eine integrative Umsetzung, eine sinnvolle Verknüpfung medienkundlicher und informatischer Kompetenzen und Inhalte, medienkompetente Lehrerkollegien, eine systematische und kontinuierliche Lehreraus- und -fortbildung sowie gute Rahmenbedingungen.

1 Einführung

Wer sich heutzutage mit Medien befasst, nutzt häufig Computer. Dies ist einer der Gründe für die Fortentwicklung des Thüringer Medienkunde-Konzeptes in den Jahren 2009/10. Neben den bekannten Themen, die schon in der Handreichung von 2001 für den Kurs der Klassenstufen 5 bis 7 enthalten waren, werden informatisches Basiswissen, der Jugendmedien- und Datenschutz sowie das Urheberrecht stärker als bisher betont ([Th09]; [LM08]; [GI08]). Künftig ist der Kurs Medienkunde auch für die Klassenstufen 8 bis 10 vorgesehen. Die Umsetzung der neuen Konzeption hat im Schuljahr 2009/10 an den weiterführenden Schulen begonnen. In diesem Aufsatz werden die neue Konzeption und die begleitende Fortbildung beschrieben und wesentliche Ergebnisse einer Lehrerbefragung zusammengefasst. Konsequenzen werden abgeleitet.

Der Kurs Medienkunde wird integrativ in den Unterrichtsfächern unterrichtet (derzeit am häufigsten in den Fächern Deutsch, Kunsterziehung, Mathematik, Englisch und Geografie). Medien, Medieneinsatz in der Schule, Computer und Informatik sind inzwischen derart verwoben, dass sich ein Zusammenführen dieser Bereiche nahezu aufdrängt. Thüringen hat sich für einen Kurs Medienkunde entschieden. Für ein Pflichtfach „Informatik“ oder „Informatik und Medien“ in der Sekundarstufe I gibt es auch weiterhin gute Argumente ([GI08], [Fo10], [Br10]).

2 Das Medienkundekonzept von 2009

2.1 Veränderungen am Kursplan Medienkunde

Der Kurs Medienkunde soll eine grundlegende Medienbildung durch das Lernen mit und über Medien absichern. Es geht um die Frage: Welche anwendungsbereiten Kenntnisse, Fähigkeiten und Fertigkeiten sollen Schülerinnen und Schüler am Ende der Klassenstufen 9/10 erworben haben, um als medienkompetent zu gelten? Neu beim Kursplan von 2009 im Vergleich zu 2001 ist die Durchgängigkeit von der Klassenstufe 5 bis 10. Abgestimmt auf die derzeitige Weiterentwicklung der Thüringer Lehrpläne (Sekundarstufe I) werden im Kursplan die einzelnen Lernbereiche kompetenzorientiert beschrieben. Neu ist auch, dass im Kursplan medienkundliche und medieninformatische¹ Kompetenzen und Inhalte verknüpft werden [Th09]. Der Kursplan ist in die folgenden Lernbereiche gegliedert:

- Information und Daten,
- Kommunikation und Kooperation,
- Medienproduktion, informatische Modellierung und Interpretation,
- Präsentation,
- Analyse, Begründung und Bewertung,
- Mediengesellschaft,
- Recht, Datensicherheit und Jugendmedienschutz.

Die Lernbereiche sind untereinander vielfältig vernetzt. Ausgehend von einer Beschreibung der Kompetenzerwartungen für jeden einzelnen Lernbereich wird eine differenzierte Darstellung jeweils für zwei Klassenstufen vorgenommen (Kl. 5/6, 7/8 und 9/10). Zusätzlich werden exemplarisch inhaltliche Empfehlungen für die Umsetzung angegeben. Der Kursplan ist unter Beachtung der konkreten Bedingungen an der jeweiligen Schule zu implementieren. Zu diesem Zweck wird an jeder Schule eine verbindliche schulinterne Lehr-/Lernplanung (SILLP) erstellt. Wie die Ergebnisse der PISA-Studie 2009 zeigen, geht eine höhere Autonomie von Schulen bei der Gestaltung von Lehrplänen mit besseren Schülerleistungen einher [OE10]. Seit dem Schuljahr 2009/10 wird, beginnend mit Klassenstufe 5, der Kurs Medienkunde im Umfang von mindestens zwei Wochenstunden pro zwei aufeinander folgender Schuljahre unterrichtet. Für bestimmte Zeiträume wird jeweils ein sogenanntes Leitfach festgelegt, in das der Kurs Medienkunde integriert ist. Das Leitfach kann zeitweise durch eine zusätzliche Unterrichtsstunde verstärkt werden (aus dem Pool der flexiblen Stunden). Am Ende eines jeden Schuljahres erhalten die Schülerinnen und Schüler einen Medienpass, der einen Überblick über erworbene Kompetenzen und thematisierte Inhalte enthält.

¹ Zum Begriff „Medieninformatik“ vgl. [BHM09].

2.2 Fortbildungskonzept

Neben den erforderlichen Planungsarbeiten besteht eine wesentliche Aufgabe in der Qualifizierung der Kollegien an den Schulen. Der Prozess der Implementierung des neuen Kursplanes wird seit 2009 durch eine Fortbildungsreihe begleitet, die vom Thüringer Ministerium für Bildung, Wissenschaft und Kultur und dem Thillm konzipiert und organisiert wird. Ziel ist es, Lehrerinnen und Lehrer von jeder Schule einzubeziehen und diese zu befähigen, die SILLP kompetent zu entwickeln und an ihrer Schule umzusetzen. Die Fortbildungen richten sich an die Lehrerinnen und Lehrer, die an den Schulen für die Koordinierung des Kurses Medienkunde verantwortlich sind („koordinierende Lehrerinnen und Lehrer“). Den Teilnehmern wird in der Regel eine Abminderung gewährt (4 Wochenstunden). Die Fortbildungsreihe besteht aus zentralen Tagungen, regionalen Veranstaltungen und individuellen Arbeitsphasen, die teilweise lernplattform-gestützt stattfinden. Die regionalen Veranstaltungen werden von einem Fachberater für Medienpädagogik bzw. einem Mitglied der Landesfachkommission Informatik geleitet (Multiplikatoren). Thematisiert werden dabei die Medien- und die informatische Bildung, das Erstellen einer SILLP, das Arbeiten mit einer Lernplattform sowie die Erstellung, Dokumentation und Präsentation von sogenannten Impulsbeispielen. Dabei handelt es sich um Unterrichtbeispiele, die für den Erwerb von Medienkompetenz besonders geeignet sind. Die Teilnehmer an der Fortbildungsreihe sollen auch eigene Erfahrungen zu E-Learning sammeln; daher finden Präsentationen mitunter als Online-Veranstaltungen unter Nutzung einer Lernplattform statt. Ausgewählte Impulsbeispiele werden in der Mediothek des Thüringer Schulportals (siehe www.schulportal-thueringen.de) veröffentlicht. In den Fortbildungen des Schuljahres 2009/10 standen die Klassenstufen 5/6 im Mittelpunkt. In den Schuljahren 2010/11 und 2011/12 geht es vorrangig um die Klassenstufen 7/8 bzw. 9/10.

3 Evaluation

Zentrale Aspekte des neuen Medienkunde-Konzepts wurden mithilfe einer Befragung² evaluiert. Befragt wurden die Lehrerinnen und Lehrer, die im Schuljahr 2009/10 an der Fortbildung teilgenommen hatten (siehe Abschnitt 2.2). Fragen wurden zur Einstellung zum Kurs Medienkunde und zu seinem neuen Konzept gestellt. Außerdem wurden schulische Rahmenbedingungen, der Qualifikationsstand sowie Einschätzungen zur Fortbildungsmaßnahme erfasst. Im Folgenden werden ausgewählte Fragestellungen und Ergebnisse der Erhebung beschrieben.

3.1 Charakterisierung der Stichprobe

An der Befragung nahmen 234 Lehrerinnen und Lehrer teil. Diese Lehrer unterrichten zu 59,6 % an Regelschulen, zu 33,5 % an Gymnasien und zu 6,9 % an anderen Schularten. Lehrer fast aller Fächer, die an Thüringer Regelschulen oder Gymnasien unterrichtet werden, nahmen an der Fortbildung teil. Die Fächer, die die befragten Lehrer am

² Zur Erfassung und Auswertung der Daten wurden GrafStat und SPSS verwendet.

häufigsten unterrichten, sind Mathematik (112), Physik (81), Wirtschaft und Recht (52) und Informatik (42). 61,4 % der Befragten waren weiblich und 38,6 % männlich. Ein Teilnehmer war 29 oder jünger, 13 zwischen 30 und 39, 117 zwischen 40 und 49, 98 Lehrer waren zwischen 50 und 59 und zwei Umfrageteilnehmer waren 60 Jahre oder älter. Dies dürfte grob der Altersstruktur der Thüringer Lehrerschaft entsprechen. Die Befragung wurde nach der letzten Veranstaltung im Mai 2010 mit Hilfe eines Online-Fragebogens durchgeführt.

3.2 Einstellung, Stellenwert und Neukonzeption

Die Einstellung, der Stellenwert des Kurses für die Schule und die Neukonzeption wurden mit einer fünfstufigen Likert-Skala erfasst (siehe Tab. 1).

Nr.	Frage	Skalenausprägung
2.2	Welche Einstellung haben Sie zum Kurs Medienkunde?	1 – negativ bis 5 – positiv
2.3	Welchen Stellenwert geben Sie dem Kurs Medienkunde für die Schule?	1 – unwichtig bis 5 – wichtig
2.4	Wie beurteilen Sie die Erweiterung des Kurses Medienkunde bis Klasse 10?	1 – negativ bis 5 – positiv
2.5	Wie beurteilen Sie die Kompetenzorientierung des Kursplans Medienkunde?	1 – negativ bis 5 – positiv
2.6	Wie schätzen Sie die erfolgte Verknüpfung informatischer und medienkundlicher Inhalte des Kurses Medienkunde ein?	1 – negativ bis 5 – positiv

Tabelle 1: Fragen zur Einstellung, zum Stellenwert und zur Neukonzeption

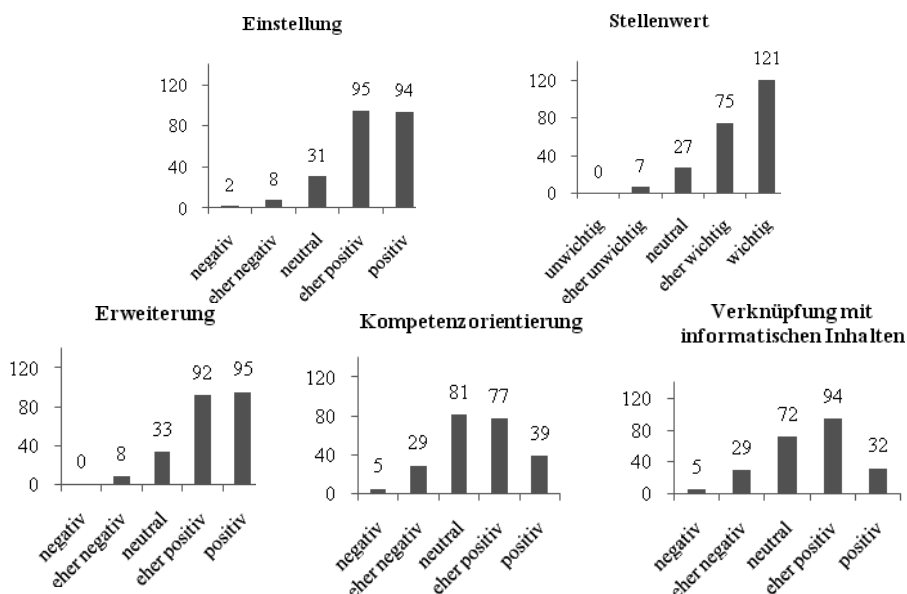


Abbildung 1: Einstellung, Stellenwert und Neukonzeption.

Die meisten koordinierenden Lehrer haben zur Medienkunde eine positive oder eher positive (79,8 %) Einstellung³. Der Stellenwert von Medienkunde in der Schule wird von 83,8 % der befragten Lehrer als wichtig oder eher wichtig eingeschätzt (siehe Abb. 1). Auch die Erweiterung von Medienkunde bis zur Klassenstufe 10 wird von den meisten koordinierenden Lehrern (82,0 %) als positiv oder eher positiv eingeschätzt. Die kompetenzorientierte Schreibweise des Kursplans (50,2 %) und die Verknüpfung medienkundlicher mit informatischen Inhalten (54,3 %) erfahren hingegen geringere Zustimmung.

Vergleicht man die Einschätzungen der Informatik-Lehrer mit denen der Nicht-Informatik-Lehrer, so ergeben sich kaum Unterschiede (siehe Tab. 2). Die Einstellung der Informatiklehrer ist etwas negativer mit einer höheren Standardabweichung. Mit Ausnahme der Bewertung der Kompetenzorientierung sind die anderen Durchschnittswerte aber etwas höher. Die höhere Standardabweichung der Informatik-Lehrer bei der Einstellung, der höhere Wert beim Stellenwert und bei der Erweiterung können im Verhältnis zweier widersprüchlicher Faktoren interpretiert werden. Möglicherweise befürworten zwar viele Informatik-Lehrer eine Ausweitung informatischer Themen, befürchten aber, dass dies mit weniger Informatikunterricht einhergeht⁴ oder dass informatische Themen weniger gut im Rahmen des Kurses unterrichtet werden.

		Einstellung	Stellenwert	Erweiterung	Kompetenz-orientierung	Verknüpfung mit informatischen Inhalten
Insgesamt	$\bar{x}$	4,18	4,35	4,2	3,5	3,42
	N	230	230	228	231	232
	SD	,856	,804	,815	,986	,953
Nicht-Informatik-Lehrer	$\bar{x}$	4,20	4,33	4,17	3,54	3,47
	N	188	189	187	190	190
	SD	,815	,804	,836	,963	,941
Informatik-Lehrer	$\bar{x}$	4,07	4,44	4,37	3,32	3,71
	N	42	41	41	41	42
	SD	1,022	,808	,698	1,083	,995

Tabelle 2: Einstellung, Stellenwert und Einschätzung der Neukonzeption.

³ Die insgesamt positive Einstellung muss allerdings differenzierter betrachtet werden, da es sich dabei um ein eher globales Konstrukt handelt. Die Antworten auf verschiedene offene Fragen erlauben ein differenziertes Bild.

⁴ An den meisten Gymnasien in Thüringen wurde Informatik als Grundkurs mit jeweils drei Wochenstunden in den Klassenstufen 11/12 angeboten. Durch die Reform der gymnasialen Oberstufe wurde die verpflichtende Stundenzahl auf zwei Stunden reduziert [Th09b].

3.3 Organisation und Rahmenbedingungen

Die schulischen Bedingungen für die Umsetzung von Medienkunde werden von den Teilnehmern der Fortbildung mit einer positiven Tendenz bei hoher Standardabweichung eingeschätzt (Mittelwert = 3,41; SD = 1,186 bei N = 231; siehe Abb. 2).

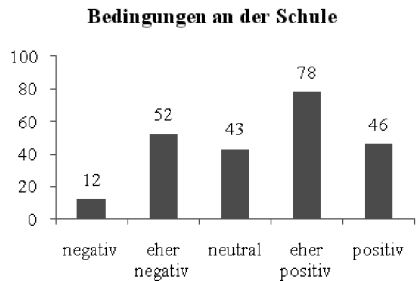


Abbildung 2: Einschätzungen zu den schulischen Bedingungen.

Die Teilnehmer konnten zusätzlich konkrete positive bzw. negative Bedingungen an ihren Schulen in offenen Antwortfeldern angeben; davon wurde rege Gebrauch gemacht (siehe Tab. 3 und 4). Probleme werden beim Ausbildungsstand bzw. bei der Motivation der Kollegen und bei der Integration in den Fachunterricht gesehen. Ein insgesamt differenziertes Bild ergibt sich bei der Ausstattung mit Computer- und Medientechnik. Neben guter Zusammenarbeit in den Kollegien wurden z.B. positive Erfahrungen bei der Beteiligung von Eltern und Fördervereinen bei der Entwicklung und Umsetzung der SILLP genannt. Ganz allgemein sollten Best-Practice-Beispiele landesweit bereitgestellt werden.

Nr.	Frage	Skalenausprägung/ Antwortformate
2.7	Wie schätzen Sie die Bedingungen an Ihrer Schule für den Kurs Medienkunde ein?	1 – negativ bis 5 – positiv
2.8	Welche Probleme sehen Sie für den Kurs Medienkunde an Ihrer Schule?	offen
2.9	Welche Bedingungen zur Durchführung des Kurses Medienkunde sind an Ihrer Schule gut erfüllt?	- gute Zusammenarbeit mit den Lehrerkollegen - gute Ausstattung mit Computer- und Medientechnik - offen

Tabelle 3: Fragen und Antwortmöglichkeiten zu schulischen Bedingungen des Kurses Medienkunde.

Bedingung	positiv	negativ
Motivation der Kollegen	22	42
Organisatorisches	14	39
Ausstattung mit Computern und Medientechnik	12	85
Klassenstärke	3	21
Motivation der Schüler	2	6
Externe Unterstützung (Eltern, Fördervereine)	2	0
Integration von medienkundlichen Themen in den Fachunterricht	1	64
Ausbildungsstand der Lehrer	0	81
Schwierigkeiten bei der Umsetzung der Fachlehrpläne (Zeitbudget)	0	11

Tabelle 4: Häufigkeit der Nennungen von positiven schulischen Bedingungen und von Problemen bei der Umsetzung des Kurses Medienkunde.

Medienkunde wurde am häufigsten in die Fächer Deutsch (189), Kunsterziehung (84), Mathematik (82), Englisch (75) und Geografie (49) integriert. Damit wird deutlich, dass Medienkunde häufig nicht im Rahmen von den Fächern unterrichtet wird, die bei den Fachkombinationen der koordinierenden Lehrer angegeben wurden. 159-mal wurde Medienkunde in bis zu vier Fächer, 71-mal in mehr als vier Fächer integriert. 136 (58,1 %) Lehrer gaben an, dass an ihrer Schule auch flexible Stunden für Medienkunde verwendet wurden. 82-mal (35 %) war das nicht der Fall. Projekttag wurden 100-mal (42,7 %) genutzt und 126-mal nicht genutzt (53,8 %).

3.4 Kompetenzeinschätzung und Fortbildungsbedarf

Die eigenen Medien- sowie die Informatikkompetenz wurden von den Fortbildungsteilnehmern mit zwei Fragen eingeschätzt. Außerdem wurde in Bezug auf die Fortbildungsmaßnahme weiterer Bedarf in drei Kategorien erfragt (siehe Tab. 5).

Nr.	Frage	Skalenausprägung/ Antwortformate
2.10	Wie schätzen Sie Ihre Kompetenzen zu medienkundlichen Themen ein?	1 – niedrig bis 5 – hoch
2.11	Wie schätzen Sie Ihre Kompetenzen zu informatischen Themen ein?	1 – niedrig bis 5 – hoch
3.8	Für welche Schwerpunkte wünschen Sie sich weitere Fortbildungsangebote? a) zu medienkundlichen Themen b) zu informatischen Themen c) zu Organisatorischem und schulinterner Lehr- und Lernplanung	offen

Tabelle 5: Fragen und Antwortmöglichkeiten zur Kompetenzeinschätzung und zum Fortbildungsbedarf.

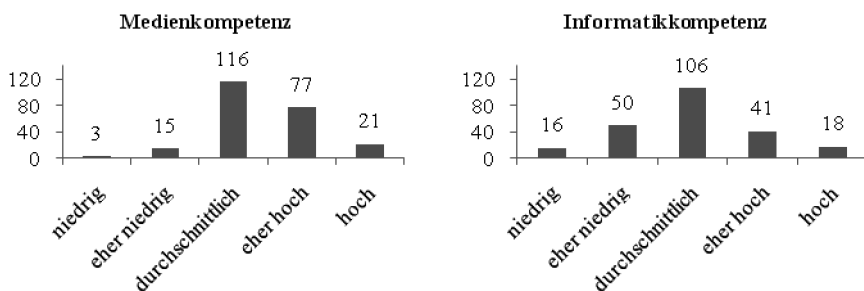


Abbildung 3: Selbsteinschätzung der Medien- und Informatikkompetenz.

Die Medienkompetenz wurde mit einem Mittelwert von 3,42 und die Informatikkompetenz etwas schwächer mit einem Mittelwert von 2,98 eingeschätzt (siehe Abb. 3). Die Streuung bei der Informatikkompetenz ist mit ,993 (bei N = 231) erwartungsgemäß höher als bei der Medienkompetenz (SD = ,797 bei N = 232). Die Kompetenzen zu medienkundlichen Themen unterscheiden sich zwischen Informatik-Lehrern und Nicht-Informatik-Lehrern kaum. Bei informatischen Themen werden die Kompetenzen allerdings mit dem Mittelwert 2,76 (SD = ,898 bei N = 190) bei Nicht-Informatik-Lehrern und mit 3,98 (SD = ,993 bei N = 41) bei Informatik-Lehrern recht unterschiedlich eingeschätzt (siehe Tab. 6)⁵. Etwas besser kann dies nachvollzogen werden, wenn der Fortbildungsbedarf analysiert wird, den die Teilnehmerinnen und Teilnehmer in offenen Antwortfeldern angaben. Bei informatischen Themen dominieren Grundlagen, Grundbegriffe und Grundkonzepte (28); Programmier- und Auszeichnungssprachen; Technisches, Betriebssysteme und Administration (je 25) sowie der Umgang mit Anwenderprogrammen (12). Bei medienkundlichen Themen nannten die Teilnehmer vor allem das Web 2.0 (63); Medienproduktion und -manipulation (54); Didaktik der Medienkunde sowie den Umgang mit Anwenderprogrammen wie Word (je 34). Nennungen zu informatischen Themen gab es insgesamt 139, zu medienkundlichen Themen 201.

		Medienkompetenz	Informatikkompetenz
Insgesamt	$\bar{x}$	3,42	2,98
	N	232	231
	SD	,797	,993
Nicht-Informatik-Lehrer	$\bar{x}$	3,34	2,76
	N	190	190
	SD	,743	,898
Informatik-Lehrer	$\bar{x}$	3,81	3,98
	N	42	41
	SD	,797	,993

Tabelle 6: Vergleich der Kompetenzeinschätzung von Informatik- und Nicht-Informatik-Lehrern.

⁵ Die Ergebnisse müssen unter verschiedenen Einschränkungen betrachtet werden. Die befragten Lehrer wurden zu medienkundlichen Themen fortgebildet und an der Schule möglicherweise aufgrund von besonderer Technik- oder Informatikaffinität als koordinierende Lehrer ausgewählt. Die Gruppe der Informatik-, Mathematik- und Physik-Lehrer ist innerhalb der Teilnehmer an der Fortbildung überrepräsentiert. Zudem ist nicht klar, welches Informatikkonzept der Einschätzung der eigenen Informatikkompetenz zugrundeliegt.

4 Resümee und Ausblick

Die Lehrerbefragung ergibt ein differenziertes Bild. Insgesamt gesehen nehmen die koordinierenden Lehrerinnen und Lehrer die 2009 erfolgten Änderungen am Medienkunde-Konzept durchaus positiv wahr. Die Kompetenzorientierung und die Verknüpfung von medienkundlichen und medieninformatischen Kompetenzen und Inhalten haben ihre Bewährungsprobe an den Thüringer Schulen jedoch noch nicht bestanden. Dazu ist auch weiterhin beharrliche Arbeit bei der Implementierung des neuen Konzepts an jeder einzelnen Schule erforderlich. Organisation und Rahmenbedingungen sind zu optimieren (siehe Abschnitt 3.3). Die Lehrerbefragung gibt dazu konkrete Hinweise.

Auch wenn sich nicht ohne Weiteres von den koordinierenden Lehrerinnen und Lehrern auf alle schließen lässt, die Medienkunde an den Schulen unterrichten, so konnten aus den Ergebnissen der Befragung dennoch wichtige Erkenntnisse gewonnen werden. Es ist davon auszugehen, dass die koordinierenden Lehrerinnen und Lehrer in der Regel auch selbst Medienkunde unterrichten und dass sie die konkrete Situation in ihren Schulen einschätzen können. Die Selbsteinschätzung der Medien- und Informatikkompetenz deutet auf weiteren Fortbildungsbedarf hin. Vor allem die vergleichsweise niedrigen Werte bei der Informatikkompetenz der Nicht-Informatik-Lehrer sind in dem Zusammenhang zur Kenntnis zu nehmen. Wie die TEDS-M-Studie 2008 [BLK08] zeigt, sind gut ausgebildete Mathematik-Lehrer eine Voraussetzung für guten Mathematikunterricht. Schüler von Nicht-Fachlehrern schneiden demnach schlechter ab. Eine ähnliche Aussage (Zusammenhang zwischen Lehrer-Qualifikation und Schüler-Kompetenzen in Mathematik bei PISA 2003) lässt sich aus der COACTIV-Studie [Kr08] ableiten. Hinzuweisen ist aber auch darauf, dass nicht jeder beteiligte Lehrer alle Aspekte des Kursplans Medienkunde gleichermaßen gut beherrschen muss. Hier gilt es an den Schulen sinnvolle Formen der Zusammenarbeit zu entwickeln.

Für weitergehenden Informatik-Unterricht gibt es in Thüringen die Möglichkeit eines Wahlfachs Informatik (künftig auch eines Wahlpflichtfachs Informatik), das bisher jedoch nicht flächendeckend an den Regelschulen und Gymnasien angeboten wird [Th01].

Aus der Lehrerbefragung lässt sich nicht ableiten, wie der neue Kursplan im Unterricht auch wirklich umgesetzt wird. Dazu wären weitergehende Untersuchungen erforderlich. Fragen der Organisation sowie Zuordnung einzelner Themen zu bestimmten Unterrichtsfächern an den Schulen könnten durch eine Analyse der SILLP oder mit der Erhebung qualitativer Daten von Lehrern beantwortet werden. Es wäre auch interessant zu erfahren, wie sich die Einstellung der Lehrerinnen und Lehrer in den nächsten Jahren ändert, wenn weitere Erfahrungen mit dem Kurs, den Inhalten und der Integration in den Fachunterricht gemacht worden sind. Des Weiteren könnte untersucht werden, welche Auswirkungen der Kurs Medienkunde auf den Informatikunterricht in den Sekundarstufen I und II hat.

Literaturverzeichnis

- [BHM09] Butz, Andreas.; Hußmann, Heinrich; Malaska, Rainer: Medieninformatik – Eine Einführung, Pearson, München 2009.
- [BKL08] Blömeke, Sigrid; Kaiser, Gabriele; Lehmann, Rainer: TEDS-M 2008 Sekundarstufe I: Ziele, Untersuchungsanlage und zentrale Ergebnisse. In: (ders. Hrsg.): TEDS-M 2008. Professionelle Kompetenz und Lerngelegenheiten angehender Mathematiklehrkräfte für die Sekundarstufe I im internationalen Vergleich, Waxmann, Münster; New York; München; Berlin, 2008; S. 11-38.
- [Br10] Breier, Norbert: Informatische Bildung und Medienbildung im Fächerkanon. In: Meyer, T.; Appelt, R.; Schwalbe, Ch.; Tan, W.-H. (Hrsg.): Medien & Bildung. Institutionelle Kontexte und kultureller Wandel. Wiesbaden: VS-Verlag 2010, S. 255-264.
- [Fo10] Fothe, Michael: Kunterbunte Schulinformatik - Ideen für einen kompetenzorientierten Unterricht in den Sekundarstufen I und II. LOG IN Verlag, Berlin 2010.
- [GI08] Gesellschaft für Informatik (Hrsg.): Grundsätze und Standards für die Informatik in der Schule. Bildungsstandards für die Informatik in der Sekundarstufe I. In: LOG IN (Beilage), 28. Jg., Heft 150/151.
- [Kr08] Krauss, Stefan; Neubrand, Michael; Blum, Werner; Baumert, Jürgen; Brunner, Martin; Kunter, Mareike; Jordan, Alexander: Die Untersuchung professionellen Wissens deutscher Mathematik-Lehrerinnen und -Lehrer im Rahmen der COACTIV-Studie. In: Journal für Mathematik-Didaktik (JMD), 29 (3/4), 2008; S. 223-257.
- [LM08] Länderkonferenz MedienBildung (Hrsg.): Kompetenzorientiertes Konzept für die schulische Medienbildung Positionspapier der Länderkonferenz MedienBildung, 2008.
- [OE10] OECD (Hrsg.): PISA 2009 Ergebnisse: Zusammenfassung, 2010.
- [Th01] Thüringer Kultusministerium (Hrsg.): Lehrplan für die Regelschule und das Gymnasium. Wahlfach Informatik/ Wahlunterricht Informatik, 2001.
http://www.thillm.de/thillm/pdf/lehrplan/rs_gy_wahl_if.pdf
- [Th09] Thüringer Ministerium für Bildung, Wissenschaft und Kultur (Hrsg.): Medienkunde, 2009.
http://www.thueringen.de/imperia/md/content/tmbwk/bildung/information/medienkunde_dez09_endfassung.pdf
- [Th09b] Thüringer Ministerium für Bildung, Wissenschaft und Kultur (Hrsg.): Informationen zur gymnasialen Oberstufe, 2009.
http://www.thueringen.de/imperia/md/content/tkm/neue_gymnasiale_oberstufe_stand_23_oktober.2007-1.pdf

Zur Diskussion von Kontexten und Phänomenen in der Informatikdidaktik

Ira Diethelm und Christina Dörge
Carl von Ossietzky Universität
Fakultät II – Department für Informatik
Didaktik der Informatik
26111 Oldenburg
{ira.diethelm|christina.doerge}@uni-oldenburg.de

Abstract: Die Diskussion über mehr Lebensweltbezug im Unterricht ist mit „Informatik im Kontext“ seit ein paar Jahren in der Informatikdidaktik im Gange. Dieser Artikel begibt sich auf die Suche nach Kriterien für eine gute InIK-Unterrichtsreihe und möchte einen Beitrag zur aktuellen Debatte liefern, indem die Begriffe „Kontext“ und „Phänomen“ diskutiert und in Bezug zueinander gesetzt werden. Dafür werden neben Veröffentlichungen aus der Didaktik der Informatik auch solche aus anderen Fachbereichen herangezogen, die sich mit den Themen „Lernen im Kontext“ und „Phänomen“ auseinandergesetzt haben. Es entsteht so ein Fragenkatalog, der bei Einordnung und Planung von kontextorientierten Unterrichtseinheiten helfen soll.

1 Einleitung

Die PISA-Studie aus dem Jahr 2000 diagnostizierte den naturwissenschaftlichen Fächern eine mangelnde inhaltliche Vernetzung sowie ein geringes Interesse der Schülern an ihnen. Dies löste einen Schock aus, welcher zur Entstehung der Kontext-Projekte Chemie im Kontext (ChiK), Physik im Kontext (piko) und Biologie im Kontext (bik) führte. Sie verfolgen das Ziel bei den Schülern mehr Interesse durch eine stärkere Verknüpfung des Unterrichts mit der Lebenswelt der Schüler zu erzeugen. Lernmotivation und Interesse werden hauptsächlich durch den persönlich wahrgenommenen Bedeutungsgehalt des Gegenstandes und die Wahrnehmung eines Kompetenzzuwachses durch Unterricht erzeugt, vgl. [Pre95]. Daran knüpft die Idee an Unterricht an für Schüler bedeutsamen Kontexten auszurichten, die ca. seit 2008 auch als Informatik im Kontext in unser Fach Einzug gehalten hat.

Viele Informatiklehrer sind fachlich leider weniger als ihre Naturwissenschaftskollegen in der Lage, geeignete Kontexte für den Unterricht aufzubereiten. Dies liegt oft am Mangel von zeitlichen Ressourcen, oder an der Tatsache, dass Informatik immer noch häufig von nicht entsprechend ausgebildeten Lehrern unterrichtet wird. Daher ist es unseres Erachtens die Aufgabe der Informatikdidaktik, fundierte Vorschläge und Kriterien für die Auswahl solcher Kontexte anzubieten, diese zu untersuchen und für die Arbeit mit ihnen nützliche Kriterien, Definitionen und Unterschiede aufzuzeigen.

In diesem Artikel wollen wir deshalb zunächst einen Blick in die Chemiedidaktik und Linguistik werfen, wo bereits andere eine Antwort suchten zur Frage „Was ist ein (guter) Kontext?“. Darauf folgend werden wir uns dem Thema „Phänomene“ zuwenden und aufzeigen, warum nach unserer Auffassung diese Sichtweise eine wichtige und notwendige Bereicherung für Informatik im Kontext ist. Dazu wird auch die Arbeit von Martin Wagenschein herangezogen, der die Bedeutung der Phänomene für die Physikdidaktik hervorhob. Daraus schnüren wir ein Paket von Fragen, deren Beantwortung bei der Planung und Analyse von kontextorientiertem Informatikunterricht helfen soll.

2 Informatik im Kontext

Die Idee, Informatik in Kontexten (IniK) und ausgehend von der Lebenswelt der Schüler zu unterrichten, wurde bereits von mehreren anderen Informatikdidaktikern besprochen. So schlägt schon Coy in [Coy05] einige Themen vor, die dazu dienen mögen, „den Informatikunterricht aus der Falle eines platonischen Naturwissenschaftsbildes (‚Das Universum der Algorithmen‘) zu befreien und mit den Alltagserfahrungen einer lebendigen Technik im gesellschaftlichen Kontext zu vermitteln“. Analog zu den Kontextprojekten der Naturwissenschaften ChiK, piko und bik basiert laut Koubek et al. Informatik im Kontext (IniK) auf drei Prinzipien, vgl. z.B. [KSSW09]:

- Orientierung an sinnstiftenden Kontexten
- Orientierung an Standards für die Informatik in der Schule
- Methodenvielfalt

Kontexte sollten danach Ereignisse aus dem Erfahrungshorizont der Schüler einschließen und einen Bezug zur Lebenswelt der Lernenden direkt erfahrbar machen. Der Ablauf einer IniK-Unterrichtseinheit kann mit dem folgenden Schema beschrieben werden; es stellt aber nur eine mögliche, zeitliche Abfolge dar:

1. Begegnungsphase
2. Neugier- und Planungsphase
3. Erarbeitungsphase
4. Vertiefungs- und Vernetzungsphase (Dekontextualisierung)
5. Reflektions- und Bewertungsphase (Rekontextualisierung)

Andere didaktische Ansätze der Informatik begründen und wählen die relevanten Inhalte oftmals aus der Informatik heraus aus. Wenn diese gefunden sind, verbleibt man entweder in der Welt der Informatik oder richtet den Blick aus der Informatik heraus nach außen und versucht einen Bezug zu der Lebenswelt der Schüler herzustellen. Der Aufbau einer Unterrichtseinheit auf Basis des IniK-Ansatzes trägt der Lebenswelt der Lernenden auf

andere Weise Rechnung: Er geht vom Kontext, von der Lebenswelt der Schüler aus und sucht aus deren Sicht nach für sie relevanten Inhalten. Das Interesse an den Inhalten ist somit bereits vorhanden, sodass die Lernenden mit ihrer Neugier die für sie relevanten Inhalte des Informatikunterrichts so weit als möglich selbst erarbeiten und entdecken.

Somit stellt die Blickrichtung des IniK-Ansatzes seine Besonderheit im Gegensatz zu anderen didaktischen Ansätzen dar: Er ist von der Lebenswelt in die Informatik gerichtet. Diese unterschiedlichen Blickrichtungen „Wo in der Welt finde ich diesen Teil der Informatik?“ einerseits und „Welchen Teil der Informatik finde ich in diesem Teil der Welt?“ andererseits werden auch in den möglichen Planungsrichtungen für kontextorientierten Unterricht deutlich:

ausgehend von Fachkonzepten: Es werden zu Lerninhalten gezielt Kontexte gesucht, deren Hintergrund recherchiert, mit gewünschten Fachkonzepten abgeglichen,

ausgehend vom Kontext: inhaltlich recherchieren, enthaltene Fachprinzipien herausarbeiten, entscheiden, welche eher im Vordergrund stehen.

In beiden Fällen bedeutet das, dass Ereignisse aus der Lebenswelt der Schüler für den Informatikunterricht aufgegriffen werden. Der Unterricht bekommt so eine Relevanz und ist für die Schüler nachvollziehbar und zusammenhängender, siehe hierzu u.a. auch [PV09], [EP10a] und [EP10b].

Nachfolgend soll nun ein theoretisch und geschichtlich fundierter Versuch unternommen werden zu klären, was ein Kontext ist, welche Arten von Kontexten sich unterscheiden lassen und welche Gütekriterien es für einen Kontext geben könnte.

3 Definition guter Kontexte

Das Wort „Kontext“ stammt vom lateinischen Verb „contextere“ und bedeutet „zusammen weben“ bzw. „Beziehung“. Allerdings ist es unmöglich, wie Gilbert in seinem Kontext-Artikel [Gil06] schreibt, eine präzise Definition zu liefern. Er bezieht sich dabei auf die Linguisten Duranti und Goodwin (ebd., S. 960). So sieht er die Funktion von Kontext darin, solche Umstände zu beschreiben, die Worten, Phrasen und Sätzen eine Bedeutung zuordnet (ebd.). Als Bedingung für einen guten Kontext fordert Gilbert: *„A context must provide a coherent structural meaning for something new that is set within a broader perspective.“*

Koubek et al. sind ähnlicher Ansicht, wenn sie einen *„Kontext als Menge von lebensweltlichen Themen bzw. Fragestellungen, die von den Schülerinnen und Schülern als zusammenhängend geordnet werden und die dadurch sinnstiftend auf deren Handlungen wirken“* betrachten, [KSSW09].

Die Linguisten Duranti und Goodwin geben vier Kriterien an, die ein guter Bildungskontext besitzen sollte (nach [Gil06], S. 960, gekürzte Übersetzung durch die Autoren):

- K1 Eine Situation mit einem gesellschaftlichen, räumlichen und zeitlichen Rahmen, innerhalb derer die näher zu untersuchenden Ereignisse auftreten,
- K2 ein Handlungsrahmen der Beteiligten, in dem die Situation mit diesen Ereignissen eingebettet ist,
- K3 der Gebrauch einer bestimmten Fachsprache über diese Ereignisse,
- K4 eine Beziehung zu nicht situationsspezifischem Hintergrundwissen.

Daraus leitet Gilbert mit Bezug auf Lerntheorien die folgenden vier Kriterien für eine gute kontextorientierte Unterrichtsorganisation (Setting) ab ([Gil06] S. 961):

- S1 Die Schüler müssen das Setting als sozialen, räumlichen und zeitlichen Rahmen wahrnehmen und die Teilnahme daran wertschätzen. Der Rahmen muss deutlich aus dem alltäglichen Leben der Schüler stammen oder von aktueller gesellschaftlicher Bedeutung sein.
- S2 Der Unterricht muss einen klar definierten Handlungsrahmen enthalten, in dem die Handlungen, die die Schüler durchführen, sowohl von Lehrern als auch den Schülern als wichtig und wertvoll wahrgenommen werden.
- S3 Die Schüler sollen im Unterricht zur Verwendung einer stimmigen Fachsprache angeregt werden. Hierzu ist wichtig, dass der Lehrer das Vorwissen der Schüler kennt.
- S4 Die Schüler müssen jedes Ereignis dieses Kontextes auf relevantes Hintergrundwissen zurückführen, das produktiv auf ihrem Vorwissen aufbaut und auf andere Kontexte und Situationen übertragen wird.

In den Kriterien S3 und S4 wird also indirekt durch den Verweis auf Fachsprache und Hintergrundwissen und die Übertragung auf andere Situationen ein Bezug zu Bildungsstandards und Kompetenzen hergestellt. Anhand dieser Merkmale lässt sich auch in Bezug auf andere Aspekte die Qualität eines kontextorientierten Unterrichts prüfen. Im Weiteren möchten wir sie nun für die Informatik adaptieren, um vier unterschiedliche Arten an Kontexten zu unterscheiden und die Unterrichtsorganisation (vgl. [Gil06] S. 966ff) zu verbessern.

4 Vier Arten von kontextorientiertem Unterricht

4.1 Kontext als Anwendung von Konzepten

Eine erste Annäherung an den Begriff „Kontext“ ist die Anwendung von Konzepten oder den Auswirkungen dieser Konzepte, um ihre Nutzung und Bedeutung hervorzuheben. So wird z. B. eine Situation aus dem Leben der Lernenden herangezogen, um darauf die abstrakten Konzepte, die zuvor gelernt wurden, anzuwenden, sodass die Lernenden besser

verstehen, worum es geht (vgl. [Gil06], S. 966f). Nachdem also z. B. ein bestimmter Sortieralgorithmus entwickelt, durchdrungen und verstanden wurde, könnte man fragen, ob man ihn gut zum Sortieren von Spielkarten verwenden könnte. Hier ist der Kontext nur eine nachträgliche Illustration. Eine Einbettung des Lerninhalts in einen Handlungsrahmen der Schüler von Anfang an findet nicht statt, somit wird nicht an diesem Kontext gelernt, was zur Folge hat, dass alle Kriterien nicht zutreffen.

4.2 Kontext als innerfachliche Wechselwirkung zwischen Konzept und Anwendung

Hierbei existiert der Kontext nur in der Vorstellung des Lernalers. Die Bedeutung wird erzeugt, indem der Fokus auf bestimmte relevante Aspekte der Struktur von informatischem Wissen gelenkt wird. Gemeint sind innerinformatische Kontexte, die zwar im Handlungsrahmen eines Informatikers in der Wirtschaft oder Wissenschaft auftreten, nicht aber – bei dieser Unterrichtsorganisation – im Handlungsrahmen der Schüler.

Diese Sicht kann als Kontext verschiedene Algorithmen anbieten, bei dem die Motive sich damit zu befassen einem Informatiker sofort einleuchten, dem Schüler hingegen nicht immer klar ist, warum er dies lernen muss. Der Kontext eines Experten wird nicht automatisch zum Kontext des Schülers. Das Kriterium S1 ist somit nicht erfüllt. Die Kriterien S2 bis S4 hingegen schon. Sollten die dabei verwendeten Begriffe für Experten sogar etwas anderes bedeuten als in der Umgangssprache (ein Beispiel aus der OOM: Instanz), kann dies zur Verwirrung bei Schülern und Lehrern gleichermaßen führen (vgl. [Gil06] S. 967f).

4.3 Kontext als mentales Konstrukt

Bücher, Zeitungsartikel, geschichtliche Beschreibungen usw. über Ereignisse mit informatischem Hintergrund können als Kontexte, als mentales Konstrukt gewertet werden. Hier sind drei Aspekte von Wichtigkeit: *Situationen* („Situations“), *Kontexte* („Contexts“) und *Erzählungen* („Narratives“). Die Situationen sind hierbei die vorher beschriebenen Aspekte, in denen die zu betrachtenden Ereignisse stattfinden. Die Kontexte entstehen durch die Transformation der Situationen durch die persönliche mentale Aktivität. Erzählungen sind die Verbindungen zwischen den Kontexten und der Lebenswelt des Lernalers (vgl. [Gil06] S. 968f).

Bei der Behandlung geschichtlicher Ereignisse, wie z. B. dem Volkszählungsurteil von 1983, können die Kriterien S1, S3 und S4 erfüllt werden. Kriterium S2 allerdings selten, weil die Schüler hier in der Regel keine Handlungen ausführen, sondern sich eher in die Rolle agierender Personen versetzen oder die Auswirkungen der historischen Ereignisse auf ihr heutiges Alltagsleben diskutieren.

4.4 Kontext als sozialer Sachverhalt

Bei dieser Sichtweise ist die soziale Dimension essenziell: Ein Kontext wird hierbei als „Kulturelle Entität in der Gesellschaft“ gesehen. So bezieht sich „Kontext“ auf Themen und Aktivitäten von Personen, welche in der Gesellschaft als wichtig angesehen werden.

Dieses Modell basiert im Wesentlichen auf „Situierem Lernen“ und der Äktivitätstheorie“ (vgl. [Gil06] S. 969f).

Gut geeignet sind hierfür Entwicklungen mit weitreichenden Auswirkungen wie das Internet. Je nachdem ob der Kontext eher als soziale Umgebung oder Handlungsrahmen angesehen wird, fallen die Kriterien S1 und S2 unterschiedlich stark aus. In dieser Art von Kontext lernen die Schüler u.a., indem sie im Kontext selbst arbeiten oder einen Teil des Kontexts (nach)erfinden. Dies könnte z. B. auf ein Softwareprojekt zutreffen, in dem etwas entsteht oder modifiziert wird, das in der Schule anschließend tatsächlich Anwendung findet und somit tatsächlich Einfluss auf das tägliche soziale Leben hat. Dann wären alle vier Kriterien erfüllt.

Eine Grenze, auf deren einen Seite die schlechten und auf deren anderen Seite die guten der beschriebenen Kontexte für den Unterricht zu finden sind, ist schwer zu ziehen. Dies ist auch nicht unsere Absicht. Eine Reflektion, zu welchem Typ von Kontext eine Unterrichtseinheit gehört, sollte aber auf jeden Fall zur Steigerung der Qualität des Unterrichts beitragen. Wir verstehen die angegebenen Kriterien und Typen eher als Werkzeug und Hilfe zur Fokussierung beim Organisieren von und Nachdenken über kontextorientierten Informatikunterricht. Zu diesem Zweck soll auch als weiteres Werkzeug das *Phänomen* herangezogen werden, bevor wir beide Ansätze zusammenführen.

5 Verbindung von Kontexten und Phänomenen

Wenn wir uns die obigen Kriterien für Kontexte ansehen, ist in K1, K2 und K3 von Ereignissen die Rede und damit indirekt auch immer von Phänomenen, denn Phänomene sind erfahrbare oder beobachtbare Erscheinungen, die in der Lebenswelt der Schülern stattfinden. So definiert z. B. der Schüler-Duden für Philosophie Phänomene als (vgl. [SDP85] S. 308) *„[d]ie Erscheinung eines Gegenstands der Außenwelt in den Sinnen. Das Phänomen nimmt eine vermittelnde Stellung zwischen den Gegenständen draußen in der Welt und dem Denken ein.“*

Die philosophische Diskussion über Phänomene, was sie sind und wie sie zu definieren sind, ist eine sehr alte : So hatten sich bereits Größen wie Anaxagoras, Plato, Aristoteles, Plutarch und Descartes damit beschäftigt und sind teilweise zu unterschiedlichen Ansichten und Definitionen gekommen [RG89]. Während in der Antike der Begriff überwiegend im Sinne von „ans Licht kommen, sich zeigen“ verwendet wurde, fand er auch später Gebrauch im Sinne von „Himmelserscheinung“. Das Historische Wörterbuch der Philosophie schreibt über den neuzeitlichen Gebrauch des Begriffs (vgl. ebd., S. 471): *„Gemäß dem griechischen Ursprung des Wortes bedeutet Phänomen bis heute das in sinnlicher Anschauung unmittelbar Gegebene, sich zeigende (Phaenomenon est per se manifestum senisbus)“, meist synonym verwendet mit lat. ‚apparentia‘, dtsh. ‚Erscheinung‘.“*

Für die Physikdidaktik forderte Wagenschein „Rettet die Phänomene!“ und schreibt in [Wag76] *„Begriffe, die nicht in ihrer Herkunft aus den Phänomenen [...] zustande gekommen sind, werden mißverstanden“*, und im Nachgang dazu in [Wag89], S. 110]: *„Was nicht auf den Phänomenen steht, wird nicht verstanden und deshalb schnell vergessen.“* Markus

Rehm baut darauf in [Reh06] ein vom Verstehen ausgehendes Kompetenzmodell auf, wobei er die Kompetenz „Verstehen von Phänomenen“ in der höchsten Stufe so beschreibt: *„Das Subjekt kann die Fragwürdigkeit des Phänomens erhellen bzw. erklären, indem es das Phänomen in die / in seine Welt einordnen kann. Es stellt eine Beziehung zwischen sich, dem Phänomen in der Welt her und baut hierdurch neue Strukturen auf“*.

Phänomene sind als Begriff auch bereits in der Informatikdidaktik aufgetreten: Neben den vielen Begründungen für Informatik als allgemeinbildendes Fach innewohnenden Bezügen zu Alltagsphänomenen, bezieht sich bereits Schwill in [Sch93] ebenfalls kurz auf sie: *„Fundamentale Ideen [...] ordnen und integrieren eine Vielzahl von Phänomenen“*. Humbert schreibt in seinem Buch von *„einem didaktisch zu gestaltende(m) Zugang zu informatisch bedeutsamen Gegenstandsbereichen [...] die Phänomenorientierung“*, [Hum06], S. 56]. Zusammen mit Puhlmann nutzt er in [HP04] Phänomene, um Testitems für „Literacy in Informatics“ zu entwickeln. Sie definieren sie als die Fähigkeit junger Menschen, die sog. „Phänomene der Informatik“ zu verstehen und zu erklären. Phänomene der Informatik sind für sie die Erscheinungen und Konsequenzen von Informatik, die im alltäglichen Leben auftreten. Dabei unterscheiden sie drei Arten (ebd., gekürzte Übersetzung durch die Autoren):

- P1 Phänomene, die direkt mit Informatiksystemen verbunden sind. Sie treten auf, wenn ein Informatiksystem bewusst genutzt wird, z. B. ein Mobiltelefon.
- P2 Phänomene, die indirekt mit Informatiksystemen verbunden sind. Sie treten in Alltagssituationen auf, die mit Informatiksystemen einhergehen ohne direkt wahrgenommen zu werden. Die Verbindung tritt erst deutlich hervor, wenn das Phänomen analysiert wird, z. B. an der Supermarktkasse.
- P3 Phänomene, die nicht mit Informatiksystemen verbunden sind, aber eine inhärente informatische Struktur beinhalten oder informatisches Folgern nahe legen wie Suchen und Sortieren.

Will man nun Phänomene im Sinne Wagenscheins als *„Weg zwischen den Phänomenen und der [...] Denkwelt, hin und auch immer wieder zurück“*, [Wag76], für Informatikunterricht nutzen, ist man beim Ansatz zu Informatik im Kontext angekommen. Die einem Kontext innewohnenden Phänomene können also darauf untersucht werden, welcher Art von Phänomen sie angehören und die daraus resultierende Unterrichtseinheit darauf, welche Kriterien für kontextorientierten Unterricht erfüllt werden oder nicht. Der von Humbert und Puhlmann begonnene Ansatz der Phänomenorientierung könnte also bereits als Informatik im Kontext vorhanden sein.

Koubek et al. schreiben zum Zusammenhang von Kontext und Situation (vgl. [KSSW09], S. 271): *„Zunächst einmal gehört jeder Kontext zu einer konkreten Situation, beide sind zusammen einmalig und unwiederholbar.“* Dies unterscheidet unserer Einschätzung nach Situationen und Phänomene deutlich voneinander und ist daher ein Grund für uns, den Begriff der Phänomene dem der Situation vorzuziehen: Ein Kontext besitzt i.d.R. mehrere Phänomene, die durchaus wiederholbar sind.

An dieser Stelle möchten wir die Definition von einem „informatischen Phänomen“ im Vergleich zu Humbert und Puhlmann im Sinne Wagenscheins mit Blick auf den Schüler und die oben aufgeführten Kontextkriterien ausschärfen:

Bei einem informatischen Phänomen handelt es sich um ein Ereignis, das durch automatisierte Informationsverarbeitung verursacht wird und im realen oder mentalen Handlungsumfeld der Schüler stattfindet.

Bei der Auswahl von Phänomenen für einen Kontext und der Vorbereitung einer kontextorientierten Unterrichtseinheit in der Informatik können mit diesen Definitionen und Kriterien folgende Fragen genauer beantwortet werden und damit als Leitfragen für die Unterrichtsplanung dienen:

F1 Welches informatische Phänomen wird genutzt?

F2 Welcher Art von Phänomenen (P1 bis P3) kann es zugeordnet werden?

F3 Welche Ausprägung haben jeweils die Eigenschaften K1 bis K4 des genutzten Kontexts?

F4 Welche der Kriterien für die Unterrichtsorganisation S1 bis S4 werden inwiefern erfüllt? und damit

F5 Zu welcher Art eines Kontextes gehört die Unterrichtseinheit (vgl. Kap. 4.1 bis 4.4)?

Die Beantwortung dieser Fragen trägt zur Einordnung und Qualitätssteigerung während der Planung bei. So kann es dabei dazu kommen, die Planungen grundlegend zu überdenken oder gar die Auswahl des genutzten Phänomens zu verwerfen und von vorn zu beginnen. Wenn z.B. der Unterricht das Kriterium S2 nicht erfüllt und damit das Phänomen gar nicht im Handlungsumfeld der Schüler auftritt, ist zu vermuten, dass die Schüler wohl weniger motiviert sein könnten und dies ein Hinweis darauf sein, dass der Lehrer dafür sorgen muss, dass sich die Schüler die Situation gut vorstellen können und somit zumindest den erforderlichen mentalen Handlungsrahmen aufzeigt.

6 Zur Übertragbarkeit auf die Informatik

Gilberts Artikel zu den Arten von Kontexten stammt aus der Chemiedidaktik. Zu Recht gibt es Hinweise darauf, dass Ideen aus einem Fachgebiet nicht einfach fraglos auf ein anderes übertragen werden können. Auch gibt es Artikel, die speziell auf die Eigenheiten der Informatik – im Vergleich zu den Naturwissenschaften – verweisen (siehe z. B. [EP10a] S. 105ff). Sind die Ansätze von Gilbert nun für die Informatikdidaktik daher überhaupt nutzbar?

Um etwas Klarheit in die Besonderheiten der Informatik zu anderen Fachgebieten zu geben, können exemplarisch Schubert und Schwill herangezogen werden (vgl. [SS04] S. 10):

„Informatik ist auch keine klassische Naturwissenschaft. Zwar untersucht sie Prozesse, die auch in der Natur anzutreffen sind, etwa informationsverarbeitende Prozesse, ihre wichtigsten Forschungsgegenstände (Maschinen, Algorithmen, Datenstrukturen) sind jedoch von Menschenhand geschaffen.“

Auch wenn es unbestreitbar ist, dass eine der Elternwissenschaften der Informatik die Mathematik ist, so sind beide nicht identisch. 1977 lieferte Volker Claus sieben Thesen über die Unterschiede von Informatik und Mathematik, siehe [Cla77]. Er machte damals deutlich, dass die Informatik kein reiner Abkömmling der Mathematik ist, die diese nur erweitert. Dennoch werden Phänomene auch in der Mathematikdidaktik genutzt, vgl. [Fre83].

Die Fachsystematik der Informatik ist ohne Frage aus den zuliefernden Wissenschaften entstanden. Diese Elternwissenschaften (z. B. Mathematik, Elektrotechnik, Physik und Informationstheorie) sind auch heute noch insofern zu erkennen, als dass es in der Informatik eine Einteilung in theoretische, praktische, technische und angewandte Informatik gibt. Daher wird eine Phänomen- und Kontextorientierung vermutlich noch schwieriger umzusetzen sein, als in den Naturwissenschaften, deren Fachsystematik sich oft durch die Erklärungsversuche für Phänomene entwickelt hat.

Phänomene und Kontexte aller Art, naturwissenschaftliche und andere, treten im Alltag der Schüler auf. Diese zu nutzen und die Schüler in die Lage zu versetzen, sie zu deuten und verantwortlich in ihnen zu handeln ist ein allgemeiner Bildungsauftrag, unabhängig davon, welcher Disziplin die Ursachen der Phänomene zugeordnet werden können. Der Wissenschaft Informatik ist dagegen im Unterschied zu den Naturwissenschaften zu eigen, dass sie die Phänomene nicht nur erklärt, sondern als ausgewiesenes Ziel auch Dinge konstruiert, die neue Phänomene verursachen können (z. B.: das Internet mit all seinen Facetten).

Wir der Meinung, dass in der Schule ein Großteil der Zeit auch mit dem Verstehen der Phänomene und nicht nur mit der Erschaffung neuer Phänomene verbracht werden sollte. Für beide Fälle aber lässt sich der hier vorgeschlagene Katalog an Leitfragen und die dafür genutzten Kriterien nutzen, um die genutzten Kontexte, darin enthaltene Unterrichtsinhalte und die Unterrichtsorganisation zu reflektieren.

7 Zusammenfassung

In diesem Artikel haben wir die Grundlagen von Kontext und Phänomenen analysiert und aus verschiedenen Quellen Definitionen und Kriterien extrahiert. Diese haben wir dann auf die Informatik übertragen, um sie für künftige Untersuchungen und Planungen des kontextorientierten Informatikunterrichts als Werkzeuge zur Einschätzung der Qualität solcher Einheiten bereitzustellen.

Hierzu haben wir fünf Fragen nach dem genutzten Phänomen, der Art des Kontextes und der Art der Unterrichtsorganisation aufgestellt, die auch als Reflektionshilfe für Lehrkräfte bei der Planung und Durchführung ihres Informatikunterrichts dienen sollen. Die Begriffe Phänomen und Kontext sollen so auch in der Informatik in der Schule zu mehr Schärfe bei den Konzeptionen von didaktischen Umsetzungen führen.

Der Weg, den die Diskussion von Informatik im Kontext bisher gegangen ist, zeigt deutlich, dass wir noch lange nicht am Ende angekommen sind. So hoffen wir mit diesem Artikel einen Beitrag zur Diskussion zu liefern, besonders hinsichtlich der Definition und dem Umgang mit den Begriffen „Kontext“ und „Phänomen“.

Literaturverzeichnis

- [Cla77] Claus, V.: Informatik an der Schule; Begründungen und allgemeinbildender Kern. Schriftenreihe des IDM 15, 26–44 (1977)
- [Coy05] Coy, W.: Informatik. . . im Großen und Ganzen. LOGIN Heft Nr. 136/137, 17–23 (2005)
- [EP10a] Engbring, D. und Pasternak, A.: IniK - Versuch einer Begriffsbestimmung. In: Brandhofer, G.; Futschek, G.; Micheuz, P.; Reiter, A. und Schroder, K. (Hrsg.). 25 Jahre Schulinformatik. Zukunft mit Herkunft. Tagungsband. Oesterreichische Computergesellschaft. 2010
- [EP10b] Engbring, D. und Pasternak, A.: Einige Anmerkungen zum Begriff IniK, 6. Workshop der GI-Fachgruppe DDI, Köllen Verlag, 2010
- [Fre83] Freudenthal, H.: Mathematics Education Library. Bd. 1: Didactical phenomenology of mathematical structures. Dordrecht: d. Reidel Publishing Company, August 1983. (zitiert in [Hum06])
- [GI08] Gesellschaft für Informatik: Grundsätze und Standards für die Informatik in der Schule, www.informatikstandards.de (2008), zuletzt besucht: 31.01.2011
- [Gil06] Gilbert, J.K.: On the Nature of „Context“ in Chemical Education. International Journal of Science Education, Routledge, Vol. 28, No. 9, 957–976 (2006)
- [Hum06] Humbert, L.: Didaktik der Informatik. 2te überarbeitete Auflage, Teubner Verlag, 2006
- [HP04] Humbert, L. und Puhlmann, H.: Essential Ingredients of Literacy in Informatics. In: Magenheimer, J. (Hrsg.); Schubert, S. (Hrsg.): Informatics and Student Assessment. Concepts of Empirical Research and Standardisation of Measurement in the Area of Didactics of Informatics, Köllen Druck + Verlag, 2004
- [KSSW09] Koubek, J.; Schulte, C.; Schulze, P. und Witten, H.: Informatik im Kontext (IniK) - Ein integratives Unterrichtskonzept für den Informatikunterricht. In: Zukunft braucht Herkunft. 25 Jahre INFOS - Informatik in der Schule, 2009
- [Pre95] Prenzel, M.: Zum Lernen bewegen. Unterstützung der Lernmotivation durch Lehre, Blick in die Wissenschaft 4/7 (1995), 58-66
- [PV09] Pasternak, A. und Vahrenhold, J.: Rote Fäden und Kontextorientierung im Informatikunterricht. In: Zukunft braucht Herkunft. 25 Jahre INFOS - Informatik in der Schule, 2009
- [Reh06] Rehm, M.: Allgemeine naturwissenschaftliche Bildung – Entwicklung eines vom Begriff „Verstehen“ ausgehenden Kompetenzmodells, Zeitschrift für Didaktik der Naturwissenschaften 12 (2006), 23–44
- [RG89] Ritter, J. und Gründer, K. (Hrsg.): Historisches Wörterbuch der Philosophie. Band 7: P-Q, völlig neu bearbeitete Ausgabe, wissenschaftliche Buchgesellschaft Darmstadt, 1989
- [SDP85] Schüler Duden "Die Philosophie Ein Sachlexikon der Philosophie, Duden-Verlag, 1985
- [Sch93] Schwill, A.: Fundamentale Ideen der Informatik. In: Zentralblatt für Didaktik der Mathematik 1, Band 25, ZDM, 20–31 (1993)
- [SS04] Schubert, S. und Schwill, A.: Didaktik der Informatik. Spektrum Akademischer Verlag, Heidelberg, Berlin (2004).
- [Wag76] Wagenschein, M.: Rettet die Phänomene! – Der Vorrang des Unmittelbaren. Scheidewege 6, Nr. 1, 1976, S. 76-93
- [Wag89] Wagenschein, M.: Erinnerungen für morgen. Pädagogische Bibliothek Beltz, 2. erg. Aufl., Weinheim und Basel, 1989

Die Didaktische Rekonstruktion für den Informatikunterricht

Ira Diethelm, Christina Dörge, Ana-Maria Mesaros, Malte Dünnebier

Carl von Ossietzky Universität

Fakultät II – Department für Informatik

Didaktik der Informatik

26111 Oldenburg

{ira.diethelm|christina.doerge|ana.maria.mesaros|malte.duennebier}@uni-oldenburg.de

Abstract: Wichtige Grundfragen der Didaktik der Informatik sind, in welcher Weise Informatikunterricht zu fassen ist, wie er geplant, durchgeführt oder erforscht werden sollte. Etliche fachdidaktische Ansätze versuchten bereits hierfür einen Rahmen zu schaffen und auch die *fundamentalen Ideen* von Schwill, die Bildungsstandards für Informatik der GI und unzählige Materialien und einige Bücher geben entsprechende Hinweise. Die Konstruktion von Informatikunterricht verläuft aber immer sehr individuell und bisher größtenteils unerforscht. Wir möchten hier einen Rahmen zur Entwicklung und Erforschung von Informatikunterricht vorstellen, welcher die *Didaktische Rekonstruktion* nutzt, um sich dem Informatikunterricht sowohl in der Forschung als auch in der Unterrichtsplanung strukturierter zu nähern.

1 Einleitung

Die Frage, was Informatik in der Schule sein soll, haben schon viele Ansätze versucht zu definieren. Die von der Gesellschaft für Informatik (GI) herausgebrachten Bildungsstandards [GI08] sind hierauf die aktuellste und erste umfassende, breit akzeptierte Antwort. Die *fachliche Klärung*, welche Inhalte geeignet sind, um die dort geforderten Kompetenzen zu erzeugen und vor allem in welcher fachlichen Tiefe sie unterrichtet werden sollten, ist eine wesentliche fachdidaktische Aufgabe. Dieser Klärungsprozess wird bisher aber in der fachdidaktischen Diskussion fast ausschließlich von fachlichen Aspekten bestimmt, teilweise fließen hier auch die *gesellschaftlichen Ansprüche* ans Fach ein.

Die *Schülerperspektiven* zu einem Sachverhalt sind dagegen aber bei der Planung des Unterrichts eher auf der Motivationsebene zu finden, wenn es darum geht, ein tragendes Beispiel zu verwenden, und wird meist erst im Anschluss an die fachliche Klärung betrachtet. Zur fachlichen Klärung selbst werden sie kaum genutzt.

Viele von Fachdidaktikern und Praktikern vorbereitete Materialien für den Unterricht, die mit viel Mühe erstellt wurden, schaffen den Sprung in die Praxis nicht. Sie stehen im Netz bereit, werden in Fachzeitschriften oder auf der INFOS veröffentlicht, werden aber selten genutzt. Ein möglicher Grund hierfür könnte sein, dass bei der *Konstruktion von Unterricht* und entsprechender Materialien die *Lehrerperspektive* außer Acht gelassen wird und

man evtl. davon ausgeht, dass ein guter Informatiklehrer schon in der Lage und willens sein wird, das Potential der Materialien zu erkennen und sich die nötigen Hintergrundinformationen anzueignen, um sie einzusetzen.

Im Folgenden wollen wir das Rahmenmodell der Didaktischen Rekonstruktion für die Informatik adaptieren und erweitern, um die genannten fünf Aspekte in eine Balance zu setzen und gleichzeitig die fachdidaktischen Forschungsthemen zu strukturieren und weiteren Forschungsbedarf zu identifizieren. Da laut Meyer [Mey07] S.5 (mit Bezug auf [Kla91] S. 88) Fachunterricht den Schülern „die Wirklichkeit der Welt aus einer bestimmten, durch die Tradition des Schulfachs vorgegebenen Perspektive erschließen“ soll und die Welt durch auftretende Phänomene wahrgenommen wird, nehmen wir als Fokussierungshilfe noch das Element *informatisches Phänomen* hinzu.

Nach einer kurzen Übersicht über unser Modell der Didaktischen Rekonstruktion für den Informatikunterricht schließt sich eine Beschreibung der Aufgabenbereiche der Didaktischen Rekonstruktion und die in ihnen enthaltenen Forschungsperspektiven an, bevor wir die einzelnen Punkte an einem Beispiel erläutern und mit einem Ausblick schließen.

2 Struktur der Didaktischen Rekonstruktion

2.1 Die Didaktische Rekonstruktion für die Naturwissenschaften

Das Verfahren der Didaktischen Rekonstruktion stammt von Kattmann, Duit, Gropengießer und Komorek. Sie schreiben in ihrem Grundsatzartikel [KDGK97]:

„Die Gegenstände des Schulunterrichts sind also nicht vom Wissenschaftsbereich vorgegeben, sie müssen vielmehr in pädagogischer Zielsetzung erst hergestellt, d.h. didaktisch rekonstruiert werden.“

Dazu schlagen sie ein iteratives Verfahren vor, bei dem jeder der drei beteiligten Aspekte als gleichwertig erachtet wird (ebd., S. 4):

„Bei der Didaktischen Rekonstruktion eines Unterrichtgegenstandes werden drei wechselwirkende Teile eng aufeinander bezogen: fachliche Klärung, Erfassung von Schülervorstellungen und didaktische Strukturierung.“

Sie beginnen bei der Didaktischen Rekonstruktion mit der fachlichen Klärung. Danach folgt die Erhebung von Schülervorstellungen. Diese beiden Ergebnismengen werden mit Hilfe der Didaktischen Strukturierung auf einander bezogen und iterativ aufgearbeitet. Abb. 1 zeigt das Dreieck der Didaktischen Rekonstruktion für die Naturwissenschaften nach Kattmann et al.

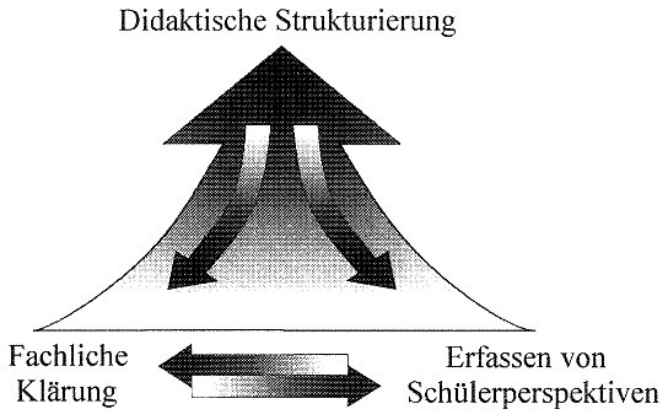


Abbildung 1: Didaktische Rekonstruktion für die Naturwissenschaften [KDGK97], S.4

Hilbert Meyer fehlt in diesem Dreieck jedoch die Variable „Lehrervorstellungen“, vgl. [MK10]:

„Diese Ausblendung hat in der Didaktik Tradition. Auch in den Allgemeindidaktischen Modellen von Klafki, Heimann/Otto/Schulz, ja selbst bei Kersten Reich wird die Variable 'Lehrer' nie als eine wirkliche Subjektivität generierender Faktor, sondern als idealisierte und hochkompetent gedachte Größe des 'Normallehrers' eingeführt.“

Gerade aber für den Informatikunterricht können wir oftmals leider nicht von idealisierten, hochkompetenten Normallehrern ausgehen.

2.2 Erweiterung des Modells für den Informatikunterricht

Da Informatik eine sehr junge Tradition als Schulfach hat, immer noch keine Einigkeit über den allgemeinbildenden Anteil von Informatik (zumindest außerhalb der GI) herrscht und dadurch auch die Lehrerbildung in Informatik in Deutschland noch sehr heterogen verläuft, erachten wir gerade für die Informatik diesen Aspekt ebenfalls für sehr wichtig. In das Modell der Didaktischen Rekonstruktion nehmen wir die Lehrerperspektive mit auf, ebenso wie die gesellschaftlichen Ansprüche ans Fach, die Meyer ebenfalls bei Kattmann et al. vermisst.

Ebenfalls im Gegensatz zu den Naturwissenschaften noch nicht etabliert ist die phänomenorientierte Sichtweise, dass Informatikunterricht das Ziel hat, den Schülern eine Sicht auf die Wirklichkeit der Welt (vgl. [Kla91]) aus dem Blickwinkel des Faches zu erschließen um damit die Phänomene des Alltag erklären zu können. In der heutigen Welt wird der Einfluss der Informationstechnologie im Alltag aber immer größer und ist immer seltener sichtbar auf Informatiksysteme bezogen. Daher nehmen wir als Fokussierungshilfe noch

das Element *informatisches Phänomen* ins Zentrum hinzu. Daraus ergibt sich Abb. 2 als unser Modell der Didaktischen Rekonstruktion für die Informatik, das wir im nächsten Abschnitt detailliert erläutern.

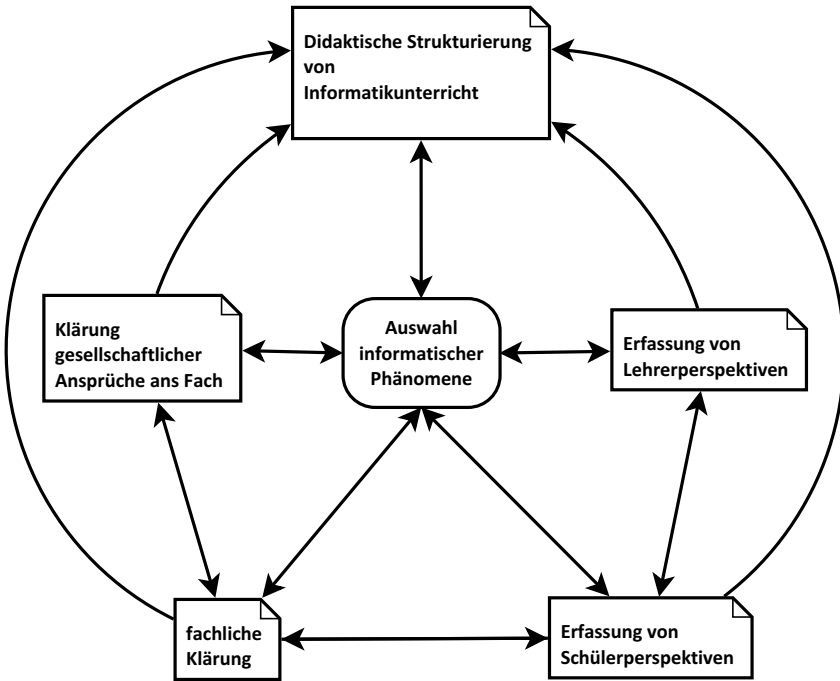


Abbildung 2: Didaktische Rekonstruktion für die Informatik

Die Pfeile geben jeweils die Richtung der Beeinflussung der verschiedenen Aufgabenbereiche an. So beeinflusst z.B. die Klärung der gesellschaftlichen Ansprüche ans Fach zwar die Didaktische Strukturierung des Informatikunterrichts, nicht jedoch umgekehrt. Nur über eine geänderte Auswahl des informatischen Phänomens, das in der Didaktischen Strukturierung Anwendung findet, wäre eine erneute, veränderte Klärung der gesellschaftlichen Ansprüche und eine erneute fachliche Klärung nötig und möglich.

3 Aufgabenbereiche der Didaktischen Rekonstruktion

3.1 Klärung gesellschaftlicher Ansprüche ans Fach

Bei der Klärung der gesellschaftlichen Ansprüche ans Fach geht es bei der Schulinformatik in erster Linie um die Frage, welchen allgemeinbildenden Gehalt der zu Rede stehende Aspekt der Informatik hat. Hier kann schon in vielen Bereichen auf Arbeiten an-

derer Fachdidaktiker der Informatik und auf die zahlreichen Empfehlungen der GI, aber auch auf Rahmenrichtlinien etc. zurückgegriffen werden, die oftmals nicht nur eine Liste von Kompetenzen und Inhalten angeben, sondern auch eine Begründung für sie enthalten. Nicht zuletzt geben die Bildungsstandards der GI hier in weiten Strecken gute Hinweise und Gründe für die Aufnahme eines Themengebiets in den Schulunterricht, meist unter dem Verweis auf die gesellschaftliche Relevanz. Aber auch aus der Diskussion um Medienkompetenz wie z.B. in [BMBF09] oder aus der JIM-Studie [MFS10] zur Mediennutzung können hier Argumentationsstränge und Auswahlhilfen entnommen werden. Diese Klärung hat dann unmittelbar einen Einfluss auf die didaktische Strukturierung eines Themas und die Auswahl der Beispiele und *informatischen Phänomene*, die durch den Unterricht erklärt und ggf. als Strukturierungshilfe für den Unterricht genutzt werden sollen.

3.2 Auswahl informatischer Phänomene

Die Einführung einer didaktischen Sichtweise, die Phänomene beinhaltet, wurde bereits von Humbert und Puhlmann angesprochen, vgl. [HP04]. Sie nutzen Phänomene der Informatik um Testitems für informatische Kompetenzen zu entwickeln. Wir nutzen sie ebenfalls für unseren Entwurf mit folgender Definition (für eine ausführlichere Diskussion und Begriffsklärung hierzu siehe [DD11], in diesem Band):

Unter einem Informatischen Phänomen verstehen wir ein Ereignis, das durch automatisierte Informationsverarbeitung verursacht wird und im realen oder mentalen Handlungsumfeld der Schülerinnen und Schüler stattfindet.

Die Auswahl informatischer Phänomene als Teil der Didaktischen Rekonstruktion von Informatikunterricht halten wir für einen wichtigen Schritt. Schon Wagenschein hat in [Wag76] gefordert, dass naturwissenschaftlicher Unterricht von den Phänomenen ausgehen sollte. Auch für die Informatik scheint dies ein guter Ausgangspunkt zu sein, da Schüler die Phänomene, die sie im täglichen Leben wahrnehmen, nicht nach dem Fachgebiet sortieren, dem die Ursachen dafür zuzuschreiben sind. Die Phänomene, die von der Informatik verursacht werden, ändern sich im Gegensatz zu naturwissenschaftlichen Phänomenen außerdem ständig. Und im Informatikunterricht sind Schüler in der Lage sie nicht nur zu erklären, sondern auch selbst die Erzeugung neuer informatischer Phänomene zu durchleben, indem sie z.B. Software erzeugen, die sie im täglichen Leben einsetzen können. Wir denken daher, dass die Auswahl informatischer Phänomene einen zentralen Aspekt der Didaktischen Strukturierung von Informatikunterricht einnimmt und sich auf alle anderen Aspekte auswirkt.

3.3 Fachliche Klärung

Unter der fachlichen Klärung versteht man in der Didaktischen Rekonstruktion in erster Linie die Inhaltsanalyse fachlicher Quellen unter Vermittlungsabsicht (vgl. [KDGK97], S. 11). Hierbei ist eine kritische Analyse der fachlichen Vorstellungen und Wirkprinzi-

prien notwendig. Kattmann et. al. weisen darauf hin, dass *„fachliche Darstellungen oft persönliche Sichtweisen enthalten, weil innerfachliche Bezüge zu kurz kommen und weil historische Verständnisse meist unreflektiert oder sogar unerkannt hineinspielen“* (vgl. ebd., S. 10).

Für die Informatik gilt dies nur bedingt, weil die Informatik keine Naturwissenschaft ist, in der die Zusammenhänge ausschließlich mit Modellvorstellungen erklärt werden müssen. Da die Zusammenhänge über die Ursachen informatischer Phänomene und die Wirkprinzipien von Informatiksystemen aus der Fachwissenschaft meist ausreichend bekannt sind, geht es in der fachlichen Klärung für die Informatik nicht so sehr darum, die „richtige“ Erklärung auszuwählen. Die Aufgabe der fachlichen Klärung in der Informatik ist vielmehr, festzulegen, in welcher fachlichen Tiefe das Thema unterrichtet werden soll und welche für den Unterricht vereinfachten Modelle der zur Rede stehenden Sache verwandt werden können. Dies hat direkten Einfluss auf die Didaktische Strukturierung des Unterrichts und auf die Auswahl der informatischen Phänomene. So kann sich bei der fachlichen Klärung durchaus ergeben, dass das zunächst favorisierte Phänomen die gewünschten Inhalte nicht ausreichend abdeckt oder umgekehrt.

3.4 Erfassen von Schülervorstellungen

Gerade die Frage nach der nötigen fachlichen Tiefe und einer geeigneten Reduktion der Fachinhalte für den Unterricht führt direkt zur Frage, welche Voraussetzungen und Vorstellungen über die genutzten informatischen Phänomene die Schüler besitzen, für die der Unterricht entworfen wird. Daher sind sie genauso wichtig wie die fachlichen Quellen. Schülervorstellungen geben wichtige Einblicke in ihre Lebenswelt und sind notwendige Anknüpfungspunkte des Lernens (siehe [KDGK97], S. 14). Dazu gehört auch, woher diese Vorstellungen stammen, eher aus fachorientierten oder lebensweltlichen Kontexten oder welche Bedeutung die genutzten Fachwörter in diesem Zusammenhang für die Schüler haben, z.B. „Instanz“ in der Justiz vs. in der Objektorientierung.

In der Informatik gibt es bisher außerhalb der Programmierung wenig Forschung zu Schülervorstellungen von bestimmten informatischen Sachverhalten, z.B. [DZ10]. Hier wird in der Zukunft eine genauere Definition, was eine Schülervorstellung in Informatik ist und eine breitere Tradition sie zu erforschen zu etablieren sein. Kattmann et al. fassen *„(u)nter 'Vorstellungen' [...] kognitive Konstrukte verschiedener Komplexitätsebenen, also Begriffe, Konzepte, Denkfiguren und Theorien, zusammen. Gegenstand der Untersuchungen sind alle von den Schülerinnen und Schülern verwendeten Vorstellungen zu einem Thema“* (siehe [KDGK97], S. 11). Diese Erhebung hat ebenfalls wieder Auswirkungen auf die fachliche Klärung und auf die Auswahl der informatischen Phänomene sowie die Didaktische Strukturierung und auch auf die Lehrerperspektiven.

3.5 Erfassung der Lehrerperspektiven

Noch unerforschter als die Schülervorstellungen sind die Lehrerperspektiven. Welche Erklärungsmuster Lehrer selbst für ein informatisches Phänomen haben, ist dabei genauso

interessant wie die Vorstellungen, die die Lehrer bei den Schülern erwarten oder wie Informatikunterricht zum ausgewählten Thema oder Phänomen strukturiert sein sollte.

Da Informatiklehrer auf unterschiedlichsten Wegen dazu kommen, Informatik zu unterrichten, sich oft Inhalte selbst angeeignet haben und wenig fachlicher Austausch oder Fortbildungsmöglichkeiten vorhanden sind, ist zu erwarten, dass die Ergebnisse dieses Bereichs der Didaktischen Rekonstruktion sehr voneinander abweichen. Zur Vorbereitung von Fortbildungen oder der Entwicklung von Unterrichtskonzepten, die in der Praxis erfolgreich sein sollen, ist die Akzeptanz der Materialien und damit die Berücksichtigung der Lehrerperspektive von sehr großem Wert. Je mehr wir auf diesem Gebiet wissen, desto größeren Einfluss hat dies auf die Didaktische Strukturierung von Informatikunterricht, der übertragbar sein sollte.

3.6 Didaktische Strukturierung

Die Didaktische Strukturierung verbindet alle vorhergegangenen Teilaspekte im iterativen und alternierenden Prozess der Didaktischen Rekonstruktion zu einem unterrichtspraktisch relevanten Forschungsergebnis. Kattmann et al. schreiben dazu (vgl. [KDGK97], S. 12): *„Grundlage der didaktischen Strukturierung ist die Verknüpfung der Ergebnisse [...] Zwischen den Konzepten, Denkfiguren und Theorien beider Seiten werden systematisch und strukturiert Beziehungen hergestellt.“* Als mögliche Produkte nennen sie folgende drei Punkte und bezeichnen sie als Formulierungsebenen der Didaktischen Strukturierung (vgl. [GK02], S. 13):

1. „grundlegende Leitlinien oder Prinzipien
2. auf empirische Ergebnisse bezogene Unterrichtskonzepte oder entsprechende Curriculumeinheiten
3. eine Reihe aufeinander bezogener Unterrichtselemente“

Die iterative Ausführung der oben beschriebenen Aufgabenbereiche kann in Bezug auf die Auswahl der informatischen Phänomene und der darauf aufbauenden Didaktischen Strukturierung zu einer Sättigung gelangen, d.h. zu einem Konzept oder Satz von Leitlinien etc. konvergieren, die anschließend in der Praxis umgesetzt und evaluiert werden sollten, bevor eine weitere Iterationsrunde beginnt und die Erfahrungen daraus einfließen.

3.7 Forschungsperspektiven

Während die Klärung der gesellschaftlichen Ansprüche ans Fach und auch die fachliche Klärung bereits zur Tradition der Didaktik der Informatik gehören, fehlen insbesondere Arbeiten zu Schülervorstellungen, informatischen Phänomenen und den Lehrerperspektiven.

Gerade die Erforschung der Schülervorstellungen hat großes Potential den Informatikunterricht stark und positiv zu verändern. Die Schülervorstellungen liefern uns wichtige

Hinweise dahingehend, womit sich die Schüler in ihrer Lebenswelt beschäftigen und was ihnen überhaupt an ihrer Umwelt auffällt. Auch geben sie Eindrücke, wie die Umwelt wahrgenommen und verstanden wird. In einer Untersuchung über Schülervorstellungen zur Funktionsweise des Internets konnte so z.B. gezeigt werden, dass einige Schüler beim Film-Streaming im Internet (z.B. auf YouTube) den Datenstrom des Filmes wie eine Schlange wahrnehmen, die bei mehreren Besuchern derselben Seite immer dünner und länger wird, weil sich nun mehrere Film-Schlangen durch eine Leitung quetschen müssen, vgl. [DZ10]. Auch das Abreißen des Datenstroms lässt sich so für die Schüler erklären, wenn man diese Schlangen wie aus Knetmasse begreift, die beim Auswalzen irgendwann so dünn werden, dass sie schließlich auch leicht abreißen.

Die Schülervorstellungen müssen auch für die fachliche Klärung genutzt werden: Die Ergebnisse aus [DZ10] zeigen, dass einige fachliche Ideen schon vorhanden sind – wenn auch nicht explizit formuliert. So ist bei der oben aufgeführten Schülervorstellung bereits so etwas wie die „sequentielle Übertragung von Paketen“ enthalten und auch das diese zerteilt und wieder zusammengefügt werden können. Auch Ansätze für das Konzept von Protokollen und der Datenübertragung allgemein sind vorhanden. So lässt sich schlussfolgern, dass diese Konzepte in die fachlichen Klärung für die Didaktische Strukturierung eines zeitgemäßen Informatikunterrichts gehören könnten.

Bei unserem Ansatz wird die Didaktische Rekonstruktion zum Analyse-Werkzeug, wo als Ausgangspunkt die gesellschaftlichen Ansprüche ans Fach und die Schülervorstellungen dienen. Für die Fokussierung verwenden wir informatische Phänomene. Diese Herangehensweise bietet u.a. den theoretischen Rahmen für die Didaktische Rekonstruktion von Kontexten auch für den Informatikunterricht im Rahmen von „Informatik im Kontext“.

4 Anwendung auf Informatikunterricht

Zum Abschluss möchten wir die Didaktische Rekonstruktion durch einmalige Iteration durch die sechs Bereiche des Modells aus Abb. 2 exemplarisch und jeweils kurz für das Thema „Internet“ durchlaufen.

Gesellschaftliche Ansprüche: In welchen Aspekten des Miteinanders spiegelt sich die Relevanz der zur Rede stehenden Sache wider?

„Jugendliche verbringen aktuell 138 Minuten pro Tag im Internet, überwiegend nutzen sie diese Zeit zur Kommunikation – meist in Communities und mit Instant Messenger. [...] Jeder Zweite zwischen 12 und 19 Jahren loggt sich täglich in seiner Online-Community ein, die meisten von ihnen sogar mehrmals täglich. [...] Zwei Drittel der jugendlichen Onliner haben Fotos oder Filme von sich ins Netz gestellt, jeder Vierte hat dort seine Instant Messenger Kontaktdaten gepostet.“ (siehe [MFS10]) Allgemein folgt daraus u.a. der Anspruch, dass Jugendliche mit dem Internet verantwortungsvoll umgehen können sollten. Dafür sollten sie die dort auftretenden Phänomene einordnen und erklären können.

Informatisches Phänomen: Was ist das beobachtbare Phänomen in der Lebenswelt der Schüler, das durch Informatik begründet ist?

Nutzbare Phänomene wären hierfür alle Ereignisse, die für Schüler im Umgang mit dem Internet auftreten. Da die Jugendlichen neben der Kommunikation das Internet meist zum Ansehen von Videos nutzen, wäre z.B. ein nutzbares Phänomen, jenes, dass ein Youtube-Video stockt.

Fachliche Klärung: Welche fachwissenschaftlichen Erklärungen gibt es hierfür?

Die fachliche Analyse nach den Ursachen für ein stockendes Youtube-Video ist so vielseitig wie das Internet selbst: Bandbreite, Hintergrundprozesse, Serverleistung, Wlan, ... Die Auswahl eines einzigen relevanten fachlichen Aspekts fällt hier sehr schwer. Dass es mit vielen Themenbereichen der Informatik verknüpft ist, zeigt das Potential des ausgewählten Phänomens.

Schülerperspektive: Welche Erklärungsmuster haben Schüler für dieses informatische Phänomen?

Hier können wir auf die Studie von Diethelm und Zumbärgel [DZ10] zurückgreifen, in der 13- und 14-jährige Schüler u.a. Youtube-Videos wie Schlangen und den Transport von E-Mails mit der Post vergleichen. Diese Vorstellungen können dann im Rückschritt wieder für die fachliche Klärung und zur thematischen Eingrenzung genutzt werden, indem man sich z.B. auf den Weg der Datenpakete und Protokolle konzentriert.

Lehrerperspektive: Welche Erklärungsmuster haben Lehrer selbst für dieses informatische Phänomen? Welche Unterrichtsziele würden sie hiermit verfolgen? Und welche Schülervorstellungen erwarten sie in ihrem Unterricht anzutreffen?

Bei den Lehrern, die wir in einer ersten qualitativen Vorstudie zu Unterrichtszielen für dieses Thema befragten, ergab sich ebenfalls ein sehr heterogenes Bild. Von Client-Server-Strukturen und dem OSI-Referenzmodell bis hin zum Aufbau eines Netzwerks oder bloßen Nutzungsanweisungen reichte das Spektrum der angegebenen Unterrichtsziele, die sie mit dem Internet verknüpfen würden. Die meisten der befragten Lehrer betonten bei der Befragung, dass sie sich mit den fachlichen Grundlagen von Netzwerken nicht besonders gut auskennen.

Didaktische Strukturierung: Wie verknüpft man die bisherigen Aspekte didaktisch sinnvoll miteinander, um daraus Unterricht zu strukturieren?

Aus allen vorangehenden Teilaspekten können Ideen für Unterricht entstehen, die die Schülervorstellungen, gesellschaftlichen Ansprüche und fachliche Klärung zusammenbringen. Unsere Forschung auf diesem Punkt ist noch nicht so weit fortgeschritten, dass wir hier ein fertiges, forschungsgeleitetes Produkt der Didaktischen Rekonstruktion anbieten könnten. Aber für das Ziel, Materialien zu produzieren, die andere Informatiklehrer verwenden sollen, schließen wir aus den wenigen Lehreräußerungen, dass die Materialien sowohl die verfolgten Unterrichtsziele als auch nötiges Hintergrundwissen der unterrichtenden Lehrern klar ausweisen sollten, um Lehrern die Entscheidung zu erleichtern, ob sie sie einsetzen werden.

5 Ausblick

In diesem Artikel haben wir das Modell der Didaktischen Rekonstruktion aus den Naturwissenschaften für die Informatik adaptiert und um für unser Fach wichtige Aspekte, die gesellschaftlichen Ansprüche und die Lehrerperspektive sowie das informatische Phänomen erweitert. Dadurch sollte ein theoretischer Rahmen aufgezeigt werden, in dem fachdidaktische Forschung, die für die Praxis nutzbare Produkte erzeugt, stattfinden kann. Bei diesem iterativen Prozess wechseln sich analytisch-hermeneutische Forschungsmethoden mit empirischen Erhebungen ab und münden in der forschungsgeleiteten Rekonstruktion von Informatikunterricht. Die Didaktische Rekonstruktion für die Informatik soll eine Hilfe sein für die Strukturierung und Evaluation von Informatikunterricht, z.B. zu Informatik im Kontext. Darüber hinaus kann sie genutzt werden, um die Lehrerbildung in Informatik, insbesondere Fortbildungen, erfolgreicher zu gestalten, da die Lehrerperspektiven einbezogen werden. Insgesamt möchten wir hiermit einen Beitrag leisten zur Steigerung der Qualität, Relevanz und Attraktivität von Informatikunterricht und den aus fachdidaktischer Forschung entstehenden Materialien.

Literaturverzeichnis

- [BMBF09] Bundesministerium für Bildung und Forschung: Kompetenzen für eine digital geprägte Kultur, Bonn, 2009
- [DD11] Diethelm, I. und Dörge, C.: Zur Diskussion von Kontexten und Phänomenen in der Informatikdidaktik. In: Informatik und Schule (INFOS) 2011, Münster, 2011
- [DZ10] Diethelm, I. und Zumbrägel, S.: Wie funktioniert eigentlich das Internet? – Empirische Untersuchung von Schülervorstellungen, In: 6. Workshop zur Didaktik der Informatik, Oldenburg, 2010
- [GI08] Gesellschaft für Informatik: Grundsätze und Standards für die Informatik in der Schule, 2008
- [GK02] Gropengießer, H., Kattmann, U.: Didaktische Rekonstruktion kurzgefasst. In: Fachdidaktik als Zentrum professioneller Lehrerbildung, diz, Oldenburger VorDrucke Nr. 387, 2. Aufl., Universität Oldenburg, 2002, S. 12-13
- [HP04] Humbert, L. und Puhlmann, H.: Essential Ingredients of Literacy in Informatics. In: Magenheimer, J. (Hrsg.); Schubert, S. (Hrsg.): Informatics and Student Assessment. Concepts of Empirical Research and Standardisation of Measurement in the Area of Didactics of Informatics, Köllen Druck + Verlag, 2004
- [KDGK97] Kattmann, U., Duit, R., Gropengießer, H. und Komorek, M.: Das Modell der Didaktischen Rekonstruktion – Ein Rahmen für naturwissenschaftsdidaktische Forschung und Entwicklung. Zeitschr.für Didaktik der Naturwiss., Jg. 3, Heft 3, 3–18 (1997)
- [Kla91] Klafki W.: Neue Studien zur Bildungstheorie und Didaktik, Beltz, Weinheim, 1991
- [Mey07] Meyer, H.: Grundformen des Unterrichts-Langfassung (Stand 26.7.2007). http://www.member.uni-oldenburg.de/hilbert.meyer/download/Grundformen_des_Unterrichts_Langfassung.pdf, zuletzt besucht 03.06.2011
- [MFS10] Medienpädagogischer Forschungsverbund Südwest: JIM-Studie 2010, <http://www.mpfs.de/fileadmin/JIM-pdf10/JIM2010.pdf>, zuletzt besucht 31.01.2011
- [MK10] Meyer, H. und Kattmann, U.: Prozesse fachdidaktischer Strukturierung, unveröffentlichtes Seminarskript, Oldenburg, 2010
- [Wag76] Wagenschein, M.: Rettet die Phänomene! – Der Vorrang des Unmittelbaren. Scheidewege 6, Nr. 1, 1976, S. 76-93

Informatiktools – Gestaltung einer Plattform für Werkzeuge für den Informatikunterricht

Ralf Romeike

Universität Potsdam
A.-Bebel-Str. 89
14482 Potsdam
romeike@cs.uni-potsdam.de

Abstract: Informatiktools sind Werkzeuge, die Informatikunterricht bei der Kompetenzvermittlung unterstützen. Vielfältige Werkzeuge, Dokumentationen dazu sowie Unterrichtsbeispiele sind bisher nur verstreut im Internet und anderen Publikationen erschienen. In diesem Paper werden die zugrunde liegenden Überlegungen zur Erstellung einer Plattform dargestellt.

1 Einleitung

Informatikunterricht orientiert sich häufig an seinen Werkzeugen, ebenso sind neuentwickelte Werkzeuge regelmäßig Gegenstand fachdidaktischer Diskussionen. Diese Werkzeuge können als Informatiktools¹ bezeichnet werden. Informatiker, Fachlehrer und Informatikdidaktiker nutzen regelmäßig ihre Fertigkeiten, in dem sie ein Tool kreieren, mit dem die Hoffnung verbunden ist, einen bestimmten Lerninhalt einfacher zu vermitteln. Vor allem Programmieren gilt als schwieriges Thema, sodass hierzu viele verschiedene Tools zur Einführung in die Programmierung existieren. Zu den frühen und bekanntesten Tools zählen *LOGO* [Pa80] und *Karel the robot* [Pa81], die bereits in den 80er Jahren eingesetzt wurden und Vorbild für verschiedene weitere Mini-Sprachen waren (bspw. Kara, Java-Hamster [RNH04; Bo05]). Moderne Programmierlernerumgebungen reichen von Tools zur Erstellung von Animationen und Spielen (z.B. Scratch [Ma04] und Greenfoot [Kö08]), zum Gestalten von interaktiven 3D-Welten (z.B. Alice [CDP00]) über Tools zur Simulation von Schwarmverhalten (z.B. Starlogo [Re96]) und Robotersteuerung (Lego Mindstorms [LSI98]) hin zu Tools zur Programmierung von Smartphone-Apps (AppInventor [Ab09]). Neben diesen Tools, die das Erlernen des Programmierens vereinfachen bzw. einer breiten Masse zugänglich machen wollen, existieren viele weitere Tools für den Informatikunterricht, z.B. Editoren (z.B. UML, Java, Prolog-, HTML-Editor), Simulatoren (z.B. JFLAP, Jeliot, AtoCC, Digitalsimulatorsimulator), Lern- und Experimentierprogramme (z.B. Infotraffic, CrypTool) und viele

¹ Tool: Werkzeug, Instrumentarium, das man für eine bestimmte Aufgabe benötigt (Duden Fremdwörterbuch).

mehr. Darüber hinaus steht eine Vielzahl freier und kommerzieller Soft- und Hardware zur Verfügung, welche den Erwerb informatischer Kompetenzen unterstützt. Gemeinsam ist allen, dass sie in ihrer Gesamtheit schwer zu überblicken und nur mit viel Aufwand für den eigenen Unterrichtseinsatz zu evaluieren sind. Guzdial [Gu04] regt an, statt weitere Tools zu entwickeln, zunächst eine Einordnung und Bewertung der vorhandenen Tools vorzunehmen. Vereinzelt finden in der informatikdidaktischen Forschung solche Bewertungen statt (bspw. im empirischen Vergleich von BlueJ und Fujaba [Do07]). Es zeigt sich aber, mit welchem Aufwand eine solche Analyse verbunden ist. Da viele Tools erfolgreich eingesetzt werden, ist es bedauerlich, dass bisher keine umfassende Plattform zur Vorstellung und Diskussion der Werkzeuge und damit verbundenen Unterrichtsmaterialien existiert. Die Existenz verschiedener Websites mit kleineren Linksammlungen zu Tools unterstreicht den Bedarf.

Im Folgenden werden grundlegende Gedanken zur Entwicklung einer Informatiktools-Plattform beschrieben, welche die angesprochene Lücke füllen soll.² Hierzu werden in Kapitel 2 die Problemlage basierend auf fachdidaktischer Forschung zu Informatiktools erörtert und existierende Plattformen hinsichtlich ihrer Struktur und Probleme beschrieben, in Kapitel 3 Taxonomien und Strukturierungskonzepte analysiert und in Kapitel 4 resultierende Gestaltungsprinzipien abgeleitet. Im abschließenden Kapitel erfolgt eine Diskussion der Erkenntnisse.

2 Forschungsstand

Regelmäßiger Gegenstand fachdidaktischer Untersuchungen sind Werkzeuge zum Programmieren lernen, vor allem mit Fokus auf die Programmieranfängerausbildung an Hochschulen: 22% aller auf dem SIGCSE-Symposium zwischen 1984 und 2003 präsentierten Paper beschrieben Tools als Hilfsmittel für Studenten oder Lehrende [Va04]. Das Kreieren eines für Informatiklehre oder -unterricht interessanten Tools ist in der Regel motiviert durch Probleme, die Anfänger mit zu komplexen Programmiersprachen, Entwicklungsumgebungen, anderer komplexer Software oder Sachverhalten haben (vgl. [PSM07]), durch eine bestimmte didaktische Idee, mit dem Ziel, die Einführung eines Paradigmas zu unterstützen oder durch übergreifende Ziele, wie z.B. der Förderung von Kreativität (bspw. Scratch). Ein weiterer Grund für die Entwicklung von Werkzeugen sind institutionelle Probleme, z.B. zur Bewältigung von zu vielen Seminarteilnehmern (vgl. [Ko06]), zur Verringerung oder Vereinfachung des Arbeitsaufwands für den Lehrenden (z.B. durch Automatic Assessment Tools) oder aktuelle Forschungsinteressen (z.B. [St06]). Trotz des mitunter großen Forschungsaufwands zur Entwicklung von Tools, werden nur wenige außerhalb ihrer Heimatinstitution verwendet [PSM07] und dies, obwohl viele kostenfrei zum Download zu Verfügung stehen. Eine Ursache hierfür liegt in der Motivation: Entwickler verlassen sich häufig allein auf die Präsentation und den (teilweise individuellen) Mehrwert eines Tools. Lehrer als möglicherweise interes-

² Dieser Artikel bildet die Grundlage zur Umsetzung des dargestellten Vorhabens. Einzelheiten zur konkreten Umsetzung können aufgrund des beschränkten Platzes an dieser Stelle nicht dargestellt werden und sollen gemeinsam mit ersten Erfahrungen zu späterer Zeit publiziert werden.

sierte Zielgruppe haben aber kaum Gelegenheit, mit dem Werkzeug in Berührung zu kommen. Auch als interessierter Informatiklehrer ist es nahezu unmöglich, einen umfassenden Überblick über aktuelle Entwicklungen zu behalten.

Verschiedene Plattformen präsentieren bereits Inhalte und Werkzeuge für den Informatikunterricht und versuchen, als „Sammelstelle“ für informatikunterrichtsspezifische Inhalte zu fungieren.

HyperForum Informatik in der Schule

Das *HyperForum Informatik in der Schule*³ beinhaltet eine umfangreiche Linksammlung zu Werkzeugen, Unterrichtsmaterialien und fachwissenschaftlichen und fachdidaktischen Artikeln *ohne nachvollziehbare Struktur*. Die Plattform wird kaum noch *aktualisiert* und *gepflegt*; viele verlinkte Dokumente sind nicht mehr erreichbar.

Zoom-Wiki

Der Bereich Informatik des *Zoom-Wiki*⁴ ist ein umfangreiches Nachschlagewerk, vermischt allerdings Unterrichtsinhalte, Enzyklopädie und Tools. Die mögliche Mitarbeit der Informatiklehrer im Wiki beschränkt sich auf wenige Personen. Ebenso scheint es keine fachspezifische *Betreuung* des Wikis zu geben. Dies führt zu einer eigenartigen Themengliederung: „Ideen für den Unterricht“, „eher Grundschule, Sek I und ITG“, „eher Sek II“, „Programmieren im Wiki“, „Informatik am WBK“, „ohne Zuordnung“, „Thema Hardware“. In jüngerer Zeit gab es *kaum Aktualisierungen*.

SwissEduc

Eine bedeutende Rolle in der Präsentation zumeist selbst entwickelter Werkzeuge für den Informatikunterricht und im Verfügbarmachen von Unterrichtsmaterialien spielt die Schweizer Plattform *SwissEduc*, welche aus dem Bildungsserver EducETH entstand. EducETH wurde 1995 als Prototyp eines Education Servers entwickelt [SE11] und war eine der ersten Quellen für Unterrichtsmaterialien im WWW [AHS96]. Die präsentierten Tools und Materialien sind qualitativ hochwertig und sind fachthematisch strukturiert. Vorgestellt werden vor allem lokal entwickelte Tools und Unterrichtsvorschläge. Eine Möglichkeit, *eigene Materialien einzustellen* oder Materialien zu *kommentieren*, existiert nicht.

Weitere Plattformen

Neben den dargestellten überregionalen Angeboten bieten verschiedene Landesbildungsserver exemplarische Unterrichtseinheiten in *variierender Qualität*. Teilweise hochqualitative Quellen stellen ebenso private und schuleigene Websites von Informatiklehrern dar, welche zumeist schulspezifisch aktuelle Unterrichtsinhalte und -sequenzen anbieten.⁵ Zu weiteren nicht weiter aktualisierten und/oder inhaltlich sehr beschränkten

³ <http://www.hyfisch.de>.

⁴ <http://wiki.zum.de/Informatik>.

⁵ Bspw. „Werkzeuge für den Informatikunterricht“ von S. Spolwig (http://www.spolwig.de/sp-hu/spolwig/einfuehrung_02/dokumente/reinhold/start.htm), Linksammlung des Gymnasiums Neustadt (<http://gym-neu.dyndns.org/~hws/werkzeuge.html>), Informatik on Stick von T. Hempel (<http://www.tinohempel.de/info/info/JoStick/index.html>).

Sammlungen gehören das life³-Projekt⁶ und Sammlungen von Schulen und Privatpersonen. Trotz dieser Fülle von Angeboten tut man sich schwer, fundierte Informationen für die Auswahl oder den Einsatz eines bestimmten Informatiktools zu finden. Viele Plattformen haben keinen *Nachschlagecharakter*, bieten *keine Diskussionsmöglichkeit*, setzen *regionale Schwerpunkte* und sind *nicht evaluiert* bzw. *referenziert*.

Zusammenfassend stellen sich folgende Punkte als problematisch dar:

- Fehlende Struktur
- Keine Aktualisierung und Pflege existierender Inhalte
- Betreuung der Plattform
- Inhalte sind nicht evaluiert bzw. referenziert
- Keine Diskussionsmöglichkeit
- Regionale Schwerpunkte der einzelnen Plattformen

3 Strukturierungs- und Bewertungskonzepte

Mit dem Vorhaben der Realisierung einer Plattform für Informatiktools stellt sich die Frage nach der Darstellung und Aufbereitung der Inhalte unter Berücksichtigung der beschriebenen Probleme. Ansätze können Taxonomien und Bewertungsversuche geben. In verschiedenen Aufsätzen wurde versucht, zumeist Programmierungsumgebungen (z.B. [GP05; KP05; AK09; Ne09]) oder Werkzeuge zur Programmvisualisierung (z.B. [My90; KPN08]) einzuordnen und zu bewerten. Mit dem Ziel, Ansätze für die Strukturierung und Bewertung von Informatiktools zu erhalten, werden die wesentlichen Aussagen im Folgenden dargestellt.

Taxonomien für Programmierlernumgebungen

Programmiersprachen werden vordergründig für Programmierer entwickelt. Die Komplexität der Werkzeuge stellt für Anfänger ein Problem dar, sich zurechtzufinden und Fehlermeldungen zu interpretieren. Kelleher und Pausch [KP05] kategorisieren Programmierungsumgebungen und -sprachen für Programmieranfänger in einer Baumstruktur, basierend auf der Frage nach der Motivation, Programmieren zu lernen: Um des Programmieren willens („Teaching Systems“) oder aus „höheren“, z.B. allgemein bildenenden, Gründen („Empowering Systems“). Diese werden hinsichtlich ihrer Merkmale weiter untergliedert:

1. Teaching Systems (helfen korrekt Programmieren zu lernen)
 - a. Mechanics of Programming
 - i. Expressing Programs
 - ii. Structuring Programs
 - iii. Understanding Program Execution
 - b. Learning Support

⁶ <http://life.uni-paderborn.de/>

- i. Social Learning
 - ii. Providing a motivating context
- 2. Empowering Systems (helfen, viele Ideen umzusetzen, ggf. auch unoptimal)
 - a. Mechanics of Programming
 - i. Code is too difficult
 - ii. Improve Programming languages
 - b. Activities enhanced by programming
 - i. Entertainment
 - ii. Education

Die Kategorien bieten interessante Anhaltspunkte, über den Einsatz eines Werkzeugs zu reflektieren. So herrscht zwar weitgehend Konsens, dass Programmierung einen wichtigen Aspekt der Schulinformatik darstellt, es ist aber strittig, ob korrektes Programmieren⁷ (1.) oder das Umsetzen möglichst vieler Ideen (2.) das primäre Unterrichtsziel darstellt. [KP05] ordnen verfügbare Tools je einer Kategorie zu. Dies ist allerdings aus zwei Gründen problematisch. Zum einen sind die Kategorien nicht trennscharf. Scratch kann bspw. verschiedenen Kategorien zugeordnet werden. Zum anderen bauen viele Tools auf den Ideen existierender Tools auf und führen auch hier zu einer Vermischung. In einem zusätzlichen Raster machen [KP05] Zuordnungen zu den Attributen eines Tools: *Style of programming, programming constructs, representation of code, construction of programs, support to understand programs, preventing syntax errors, designing accessible languages, support communication, choice of task*. Dies sind sinnvolle Attribute zur Charakterisierung von Programmierumgebungen. Es bleibt zu berücksichtigen, dass z.B. die Attributwerte von „choice of tasks“ wie *fun & motivating, useful* und *educational* subjektive Charakteristika eines Tools darstellen. [PSM07] analysieren Veröffentlichungen zu Programmierereinführungskursen und identifizieren bzgl. Tools folgende über Programmierumgebungen hinausgehende Kategorien:

- 1. Visualization Tools
- 2. Automated Assessment Tools
- 3. Programming Environments
 - a. Programming Support Tools
 - b. Microworlds
- 4. Other Tools

Alle genannten Kategorien sind auch für den Informatikunterricht relevant, decken aber nicht die Vielzahl der verschiedenartigen Tools ab. [GP06] präzisieren folgende (nicht trennscharfe) Kategorien für Programmierlernumgebungen:

- Microworlds (z.B. Logo, Karel, Alice)
- Visual programming environments (iconic and textual) (z.B. RoboLab, JPie)
- Flow Model Environments (z.B. RAPTOR)
- Object workbench environments (z.B. BlueJ, jGRASP)
- Algorithm realization environment (kinesthetic, multimedia, animation, graphics) (z.B. Lego Mindstorms)

⁷ bspw. unter Nutzung textueller Programmierung und effizientem Programmcode.

Es wird deutlich, dass allein im Bereich der Programmierwerkzeuge verschiedenste Kategorisierungen möglich sind. Eine Struktur lässt sich, auch vor dem Hintergrund der Verschiedenartigkeit der Werkzeuge, nicht ausmachen. Zur Charakterisierung eignen sich Stichworte, die z.B. in einer dazugehörigen Tag-Cloud visualisiert werden.

Bewertung von Lernumgebungen

Verschiedene Ansätze wurden entwickelt, Lernumgebungen der Informatik zu bewerten, insbesondere Programmierlernumgebungen. Gängige Methoden sind *anekdotische*, *analytische* und *empirische Überprüfung* (vgl. [GP05]). Häufig finden sich *anekdotische Angaben* zum Einsatz eines Werkzeugs. Zumeist stellt der Entwickler seine Eindrücke und Beobachtungen vom Einsatz des Tools im Unterricht dar (z.B. [Re07]). Zwar sind Anekdoten allein nicht überzeugend. Werden die Eindrücke allerdings von vielen geteilt, lassen sich hieraus Hypothesen für weitergehende Evaluation ableiten. Einer *Analyse* liegen Kriterien zugrunde, hinsichtlich derer ein Tool bewertet wird. Dies kann bspw. die Einordnung hinsichtlich o.g. Taxonomien sein, das zugrunde liegende Paradigma oder Kriterien hinsichtlich der Unterstützung des Lernens. Eine *empirische Überprüfung* erfordert das qualitative oder quantitative Auswerten von beobachteten oder erhobenen Daten im Prozess der Nutzung eines Tools, z.B. Zensuren, Fragebögen oder Interviews. Aufgrund der Nachvollziehbarkeit und Wiederholbarkeit solcher Studien stellen sie zwar eine solide Grundlage zur Bewertung von Tools dar, sind aber auch sehr aufwändig (vgl. [Do07]).

Die dargestellten Möglichkeiten der Bewertung scheinen umfassend zu sein, eignen sich in der Praxis aber nur bedingt aus folgenden Gründen. Ob sich ein Tool vielversprechend im Unterricht einsetzen lässt, hängt nicht nur von seiner didaktischen Idee, sondern von einer Vielzahl weiterer Aspekte ab: z.B. Stabilität, Verfügbarkeit von Unterrichtsmaterialien, Kosten, Erweiterbarkeit etc., die unterschiedlich stark zu gewichten sind. Dazu kommen individuelle Gründe wie Kursziele, bevorzugte Sprache und/oder bevorzugte Vorgehensweise. Einen möglichen Ausweg aus dem Dilemma stellt die Möglichkeit der Abgabe erfahrungsbasierter Bewertungen dar, welche in ihrer Gesamtheit wieder ein stimmiges Gesamtbild erzeugen könnten. Die dargestellten Kriterien können dafür als Anhaltspunkte dienen. Die Informatiktools-Plattform kann helfen, Studien zusammenzutragen und zusammenzufassen, darüber hinaus aber vor allem anekdotisch und analytisch zur Bewertung der präsentierten Tools beitragen. Die GI-Empfehlungen zu Bildungsstandards Informatik bieten einen Rahmen, durch ein Tool geförderte Kompetenzen zu kategorisieren. Vor dem Hintergrund des Potentials von Softwarewerkzeugen, Kreativität im Informatikunterricht zu fördern, bieten die Kriterien für kreativitätsunterstützende Softwarewerkzeuge im Informatikunterricht [Ro09] einen weiteren Ansatzpunkt zur Bewertung entsprechender Werkzeuge.

4 Konzeption und Umsetzung

Hinter *informatiktools.de* steckt die Idee, eine Plattform zu erstellen, welche die im WWW verteilten „Schätze“ zusammenführt und unter dem Dach der Werkzeuge er-

reichbar macht. [AHS96] charakterisierten bereits 1996 die Möglichkeiten des WWW für den Austausch von Unterrichtsmaterialien treffend: „WWW stellt eine erstklassige Plattform für den einfachen, kostengünstigen und schnellen Austausch von Unterrichtsmaterialien dar.“ Existierende Plattformen fokussieren allerdings überwiegend auf den Aspekt des „Zurverfügungstellens“. Ein weiterer genannter Aspekt wird auch dort nicht umgesetzt: „Die Benutzer können auch zu einzelnen Unterrichtsmaterialien Kommentare und Ergänzungen anbringen oder selbst Unterrichtsmaterialien auf den Server hinaufladen“. Tatsächlich hat sich mit der Entwicklung des *Web 2.0* gezeigt, wie viel Potenzial hinter der Beteiligung der Nutzer steckt. [AHS96] greifen die Metapher eines Strickwarenladens als Analogie für Fachserver auf:

„In einem Strickwarenladen kann man nicht nur Wolle und Stricknadeln kaufen. Man kann sich auch beraten lassen: Auf Wunsch werden individuelle Strickmuster angefertigt, Anpassungen oder Überarbeitungen bestehender Strickmuster vorgenommen. Der Kontakt Kunde/Geschäft ist viel enger, und es kann durchaus sein, daß eine Kundin ihre Strickwaren im Geschäft zum Verkauf anbietet. Genauso kann man sich speziell auf Schulen ausgerichtete Fachserver vorstellen.“

Aus diesem Grund soll die Plattform für *Informatiktools* es ermöglichen, Werkzeuge zu bewerten, Referenzen einzustellen, zu finden sowie zu diskutieren. Die Plattform wird betreut und gepflegt, ermöglicht aber jedem Interessierten, sowohl Entwicklern als auch Nutzern von Tools, sich an der Erstellung, Weiterentwicklung und Diskussion zu beteiligen. Der Bedarf an einer solchen Website wird immer wieder in Lehrerfortbildungen deutlich, in welchen Workshops zu Werkzeugen für den Informatikunterricht eine große Rolle spielen und stark besucht werden.

Wesentliche der erforderlichen Funktionen bietet ein moderiertes und gepflegtes Wiki: Entwickler und Nutzer eines Tools können sich an der Vorstellung bzw. Diskussion des Tools beteiligen. Wikis sind bekannt, werden von Informatiklehrern teilweise selbst eingesetzt und regelmäßig verwendet (Wikipedia). Zur besseren Erfassung der Strukturen wird auf die Möglichkeiten eines *Semantic-Wiki* [SBB06] zurückgegriffen. Initiierung und wissenschaftliche Begleitung obliegen einem Informatikdidaktiklehrstuhl und können somit auch im Rahmen der Informatiklehrerbildung durch Studierende regelmäßigen Input erhalten. Die in den GI-Empfehlungen zu Bildungsstandards Informatik beschriebenen Inhalts- und Prozessbereiche können neben den oben beschriebenen als Schlagwörter zur Charakterisierung der Tools herangezogen werden. Das Template eines *Informatiktools* strukturiert sich damit folgendermaßen: Beschreibung des Tools, Schlagwörter, Altersstufen, Voraussetzungen, Stärken/Schwächen (Kritik), Unterrichtsbeispiele, Literatur zum Tool, Wissenschaftliche Veröffentlichungen, weitere Referenzen/Quellen, Referenzen zu Institutionen/Schulen, welche das Tool verwenden, Verweise auf Ansprech- und Diskussionspartner, Nutzerbewertungen. Für jedes Tool/jede Wikiseite stellt ein Template die Struktur zur Verfügung. Entsprechend den vielen interessierten Nutzern der verschiedenen Tools scheint es realisierbar, zumindest erfahrungsbasiert eine Einschätzung der im Informatikunterricht eingesetzten Tools vorzunehmen.

5 Diskussion

Im WWW existieren verschiedene Versuche, Materialien und Ideen für den Informatikunterricht verfügbar zu machen. Probleme der Plattformen sind vor allem mangelnde Aktualität und Pflege und fehlende Möglichkeiten der Mitgestaltung. Die Praxis zeigt, dass Tools einen prägenden Einfluss auf den Informatikunterricht haben, das Beschaffen von Informationen zu Tools sich aber oft als problematisch erweist. Erfahrungen auf den Lehrerweiterbildungsveranstaltungen der Informatiktage in den einzelnen Bundesländern zeigen, dass bei den interessierten Lehrern ein hoher Informationsbedarf und großes Interesse hinsichtlich der Werkzeuge für den Informatikunterricht besteht. Eine Mehrzahl der angebotenen Workshops beschäftigt sich mit Werkzeugen und ist regelmäßig stark nachgefragt⁸. Das Problem von Lehrenden hinsichtlich neuer Tools, Aufgaben zu einem noch nicht vertrauten Tool entwerfen zu müssen, stellt oft eine Hemmschwelle für den Einsatz dar. Vermutlich liegt der Erfolg von Tools wie z.B. Kara auch darin, dass sie eine umfangreiche Sammlung an Aufgaben zur Verfügung stellen. Auch bei Visualisierungswerkzeugen zeigt sich, dass Visualisierung allein nicht ausreicht, Sachverhalte zu verstehen, sondern erst das aktive Auseinandersetzen, z.B. indem Fragen beantwortet werden, Vermutungen aufgestellt werden oder experimentiert wird, ein besseres Lernergebnis erzielt [HDS02].

Sollen (noch mehr) Tools im Informatikunterricht eingesetzt werden? Sicherlich nicht, im Informatikunterricht hat der Einsatz von Software bereits einen hohen Stellenwert. Die Praxis der Lehrerweiterbildungen zeigt, dass Lehrer regelmäßig mit dem Angebot an verschiedenen Tools überfordert sind und z.B. aufgrund von Unkenntnis oder wenig Zeit Tools einsetzen, die in ihrer didaktischen Idee überholt oder in ihrer Komplexität durch deutlich einfachere Tools ersetzbar sind. So setzen z.B. viele Lehrer *GIMP* oder *Photoshop* zur Bildbearbeitung in der Sekundarstufe 1 ein, beschränken sich dabei aber auf Bildbearbeitungswerkzeuge, die nicht die Komplexität der Software benötigen und z.B. einfacher in *Pixlr* [Pi11] verfügbar und damit für Schüler altersgemäßer sind.

Ein Mehrwert der Plattform ergibt sich damit vordergründig aus dem praktischen Nutzen: Die Vielfalt an Informatiktools und korrespondierenden Unterrichtsentwürfen kann Lehrern besser zugänglich gemacht werden. Dies gilt auch für Tools, die in den Fachdiaktiken der Hochschulen entwickelt wurden. Häufig werden diese zwar publiziert und in Fachzeitschriften veröffentlicht, erreichen aber die Unterrichtswirklichkeit nicht, da nur wenige Fachlehrer Fachzeitschriften lesen. Als Beispiele hierfür können *InfoTraffic* [AH07], *Leo* [US01] und *Puck* (außerhalb Thüringens) [Ko06] genannt werden. Darüber hinaus kann die Plattform Ideen zu vielen interessanten Unterrichtsmethoden durch den Einsatz von Tools liefern, z.B. das gegenseitigen Stellen und Lösen von Quiz- bzw. Übungsaufgaben durch Lerngruppen [DHL08] oder gegenseitiges Begutachten innerhalb von Lerngruppen [Ko06].

Eine unüberschaubare Anzahl an Informatiktools wurde entwickelt, von denen Informatikunterricht profitieren kann. Leider ist wenig über den tatsächlichen Einfluss dieser

⁸ So stehen bspw. auf dem aktuellen Berlin-Brandenburger Informatiktag in 7 der 13 angebotenen Workshops Werkzeuge im Mittelpunkt, beim SH-HILL sind es sogar 11 von 16.

Tools bekannt. Aus wissenschaftlichen Gesichtspunkten bietet die Plattform die Möglichkeiten, Entwicklungs- und Diskussionsprozesse informatikdidaktischer Forschung (Werkzeuge) und Praxis sichtbar zu machen. Zugriffszahlen, Diskussionsaktivitäten und Publikationsmöglichkeiten für neue Werkzeuge sollen für den wissenschaftlichen Diskurs herangezogen werden.

Die Eröffnung der Plattform ist zur INFOS 2011 geplant.

Literatur

- [Ab09] Abelson, H.: App Inventor for Android. In Google Research Blog, July 31, 2009.
- [AHS96] Ackermann, S.; Hartmann, W.; Stumm, M.: Unterrichtsmaterial über WWW. In LOG IN 16(1), 1996; S. 61-68.
- [AK09] Ali, A.; Kohun, F. G.: Considerations for selecting a programming language to teach perspective teachers. Proc. Alice Symposium, 2009.
- [AH07] Arnold, R.; Hartmann, W.: Pragmatische Empfehlungen zur Entwicklung von interaktiven Lernumgebungen. Proc. INFOS 2007, 12. GI-Fachtagung Informatik und Schule, 2007.
- [Bo05] Boles, D.: Spielerisches Erlernen der Programmierung mit dem Java-Hamster-Modell. In Lecture Notes in Informatics (LNI)-Proceedings 60, 2005; S. 243-252.
- [CDP00] Cooper, S.; Dann, W.; Pausch, R.: Alice: a 3-D tool for introductory programming concepts. Proceedings of the fifth annual CCSC northeastern conference on The journal of computing in small colleges. Ramapo College of New Jersey, Mahwah, New Jersey, United States, Consortium for Computing Sciences in Colleges, 2000.
- [DHL08] Denny, P.; Hamer, J.; Luxton-Reilly, A.; Purchase, H.: PeerWise. Proceedings of the 8th International Conference on Computing Education Research. Koli, Finland, ACM, 2008.
- [Do07] Dohmen, M.: Empirisches Untersuchungsdesign zum Medieneinsatz im objektorientierten Informatikunterricht Proc. Didaktik der Informatik in Theorie und Praxis, INFOS 2007, 12. GI-Fachtagung Informatik und Schule, Siegen, GI, 2007.
- [GP05] Gross, P.; Powers, K.: Evaluating assessments of novice programming environments. Proc., ACM, 2005.
- [GP06] Gross, P.; Powers, K.: Work in progress-a meta-study of software tools for introductory programming. Proc., IEEE, 2006.
- [Gu04] Guzdial, M.: Programming environments for novices. In Computer science education research, 2004; S. 127-154.
- [HDS02] Hundhausen, C. D.; Douglas, S. A.; Stasko, J. T.: A Meta-Study of Algorithm Visualization Effectiveness. In Journal of Visual Languages & Computing 13(3), 2002; S. 259-290.
- [KPN08] Kasurinen, J.; Purmonen, M.; Nikula, U.: A Study of Visualization in Introductory Programming. Proc. 20th annual Meeting of Psychology of Programming Interest Group, Lancaster, UK, 2008.
- [KP05] Kelleher, C.; Pausch, R.: Lowering the barriers to programming: A Taxonomy of Programming Environments and Languages for Novice Programmers. In ACM Computing Surveys 37(2), 2005; S. 83-137.
- [Ko06] Kohl, L.: Puck-eine visuelle Programmiersprache für die Schule. In informatica didactica 8(7), 2006.
- [Kö08] Kölling, M.: Greenfoot: a highly graphical ide for learning object-oriented programming. Proc., ACM, 2008.
- [Ko06] Koubek, J.: Das Gutachtersystem als asynchrone nicht-kooperative Lernumgebung. Proc. Grundfragen multimedialen Lehrens und Lernens, 2006.

- [LSI98] LEGO_SYSTEMS_INC.: Lego Mindstorms Robotics Invention System. 1998. <http://mindstorms.lego.com>
- [Ma04] Maloney, B., Kafai, Rusk, Silverman, Resnick: Scratch: A Sneak Preview. In IEEE Computer Society, 2004; S. 104 - 109.
- [My90] Myers, B. A.: Taxonomies of visual programming and program visualization. In J. Vis. Lang. Comput. 1(1), 1990; S. 97-123.
- [NRA02] Naps, T. L.; Rößling, G.; Almstrum, V.; Dann, W.; Fleischer, R.; Hundhausen, C.; Korhonen, A.; Malmi, L.; McNally, M.; Rodger, S.; Velazquez-Iturbide, J. A.: Exploring the role of visualization and engagement in computer science education. Working group reports from ITiCSE on Innovation and technology in computer science education. Aarhus, Denmark, ACM, 2002.
- [Ne09] Nesbit, T.: The Teaching of Introductory Programming: Issues of Context. In, 2009.
- [Pa80] Papert, S.: Mindstorms : children, computers, and powerful ideas. Basic Books, New York, 1980.
- [Pa81] Pattis, R. E.: Karel the robot: a gentle introduction to the art of programming. John Wiley & Sons, 1981.
- [PSM07] Pears, A.; Seidman, S.; Malmi, L.; Mannila, L.; Adams, E.; Bennedsen, J.; Devlin, M.; Paterson, J.: A survey of literature on the teaching of introductory programming. Working group reports on ITiCSE on Innovation and technology in computer science education. Dundee, Scotland, ACM, 2007.
- [Pi11] pixlr.com: pixlr photo editing services. 2011. <http://pixlr.com> (15.01.2011).
- [RNH04] Reichert, R.; Nievergelt, J.; Hartmann, W.: Programmieren mit Kara: ein spielerischer Zugang zur Informatik. Springer, 2004.
- [Re96] Resnick, M.: StarLogo: an environment for decentralized modeling and decentralized thinking. Proc., ACM, 1996.
- [Re07] Resnick, M.: Sowing the Seeds for a More Creative Society. Proc. Learning & Leading with Technology, International Society for Technology in Education (ISTE), 2007.
- [Ro09] Romeike, R.: Softwaretools für kreatives Lernen im Informatikunterricht. In Tagungsband zur GI-Fachtagung Informatik und Schule INFOS Berlin, 2009.
- [SBB06] Schaffert, S.; Bischof, D.; Bürger, T.; Gruber, A.; Hilzensauer, W.: Learning with semantic wikis. Proc. First Workshop on Semantic Wikis -- From Wiki To Semantics, Citeseer, 2006.
- [St06] Stechert, P.: Unterrichtsmodellentwicklung zur Förderung des Informatiksystemverständnisses mit Entwurfsmustern. Proc. Workshop der GI-Fachgruppe „Didaktik der Informatik“, 2006.
- [SE11] SwissEduc: Informationen über SwissEduc. 2011. <http://www.swisseduc.ch/about/geschichte/index.html>
- [US01] Universität Siegen: LEO - Lernumgebung für objektorientiertes Modellieren im Informatikunterricht. 2001. <http://www.die.informatik.uni-siegen.de/pgleo> (15.01.2011).
- [Va04] Valentine, D. W.: CS educational research: a meta-analysis of SIGCSE technical symposium proceedings. In SIGCSE Bull. 36(1), 2004; S. 255-259.

Was ist/kann/soll Informatikunterricht?

Mehr als nur ein Plädoyer für empirische Forschungen zum alltäglichen Informatikunterricht

Dieter Engbring

FG Didaktik der Informatik
Universität Paderborn
Fürstenallee 11
33102 Paderborn
didier@upb.de

Abstract: Nur wenige Schülerinnen und Schüler wählen das Fach Informatik im Wahlpflichtbereich der Sekundarstufe II. Die Zahlen für die Informatik sind noch viel geringer als bei den *unbeliebten Naturwissenschaften* Chemie und Physik. Dies lässt für ein Pflichtfach in der Sekundarstufe I nichts Gutes erahnen. In diesem Beitrag wird auf der Grundlage der Zahlen für Nordrhein-Westfalen und den Erkenntnissen der Didaktik der Physik und Chemie nicht nur dafür plädiert empirisch den Alltag des Informatikunterrichts zu erforschen; es werden bereits erste Forschungsansätze aufgezeigt.

1 Einleitung

Das Fach Informatik hat sich in all den Jahren, da es existiert, immer wieder gewandelt. Viele Beiträge zum Informatikunterricht beschreiben diesen Wandel. Es handelt sich – dies betrifft die INFOS als auch die LOG IN – um Praxisberichte, in denen das Gelingen von Unterrichtseinheiten dargestellt wird. Anlass solcher Berichte ist oftmals die Nutzung neuer „Werkzeuge“. Diese Berichte spiegeln jedoch nicht die alltägliche Praxis wieder, die Unterrichtsreihen finden oftmals unter besonderen Rahmenbedingungen statt, indem z.B. besonders engagierte Lehrende besonders interessierte Lernende unterrichten.

In diesem Beitrag wird dafür plädiert, dass sich die fachdidaktische Forschung dem alltäglichen Informatikunterricht widmet. Dahinter stehen zwei Vermutungen. Zum einen ist Informatikunterricht in seiner Breite wohl gar nicht so innovativ, wie dies durch die oben genannten Berichte zum Ausdruck kommt. Zum anderen scheint Unterricht, der durch technische Innovation getrieben ist (wird), wohl nur wenige zu erreichen. Daraus folgen unter Berücksichtigung der Erkenntnisse aus der Erforschung naturwissenschaftlichen Unterrichts Hinweise auf die Gestaltung des Informatikunterrichts in beiden Sekundarstufen, der dann weniger auf *Werkzeuge* und Programmierung ausgerichtet ist.

Dazu wird in diesem Beitrag der Stand fachdidaktischer Forschung in den Kontext der

bildungspolitischen Arbeit gerückt, die die Vertreter der Fachdidaktik in der vergangenen Jahren immer wieder geleistet haben und leisten mussten. Am Beispiel der Bildungsstandards wird die darin liegende Problematik der bildungspolitischen Arbeit erläutert. Danach wird anhand von Daten zum Informatikunterricht in Nordrhein-Westfalen seine prekäre Lage dokumentiert. Ein Blick über den Zaun zu den naturwissenschaftlichen Fächern, die in der Sekundarstufe II in dem selben Aufgabenfeld gewählt werden können, gibt den Blick frei auf mögliche Probleme, die ein Pflichtfach in der Sekundarstufe I evozieren würde aber auch auf erste Lösungsansätze. Zum einen wird deutlich, was mit Forschung zur alltäglichen Praxis gemeint ist und zum anderen werden inhaltliche (Um-)Orientierungen in den Blick genommen.

2 Fachdidaktik und Bildungspolitik

Erst langsam etabliert sich in der Didaktik der Informatik eine fachdidaktische Forschung, die diesen Namen auch verdient. Was auf den ersten Blick nach einem Vorwurf klingt, ist in der Tat gar nicht so gemeint. Es ist eine Beschreibung des Status Quo, für die es nachvollziehbare Erklärungen gibt. Drei wesentliche Erklärungsansätze zu den Problemen der Didaktik der Informatik (DDI) werden im Folgenden dargestellt.

1. In der DDI gibt es zu wenig Personal:

Erst in den letzten zehn Jahren sind Professuren für Didaktik der Informatik in einem nennenswerten Umfang geschaffen worden. Zuvor gab es nur einige wenige Stellen, deren Forschungsschwerpunkt zudem nicht unbedingt der Informatikunterricht war oder noch immer nicht ist. In den zehn Jahren sind auch Dissertationen entstanden, die der Didaktik der Informatik zuzuordnen sind.¹ Ein Teil der Personen, die sich auf diesem Gebiet promoviert haben, sind inzwischen zu Professoren auf Zeit oder auf Dauer berufen worden. Vielleicht ist der Höhepunkt dieser Entwicklung auch schon erreicht. Nicht über all dort, wo Informatik-Lehramtsausbildung betrieben wird, ist die wissenschaftliche Ausrichtung der Lehrenden tatsächlich fachdidaktisch. Oft steht schon von der wissenschaftlichen Herkunft der mit der Ausbildung betrauten Personen eher E-Learning im Fokus ihres Interesses. Es ist hier in diesem Beitrag nicht der Ort ausführlich über die Gründe zu spekulieren, warum Professuren, die ursprünglich für Informatik in der Schule ausgeschrieben werden, dann mit Forschern zum E-Learning (wieder-)besetzt werden. Es liegt die Vermutung nahe, dass die zu erwartenden höheren Drittmittel im Bereich E-Learning diese Entscheidungen mit beeinflussen. Für Unterrichtsforschung oder auch fachdidaktische Forschung im allgemeinen ist es schwer, Drittmittel einzuwerben.

2. DDI ist zu jung:

Durch die Ausführungen zu 1. scheint der zweite Erklärungsansatz widerlegt zu sein. Die Jugend des Fachgebietes Didaktik der Informatik spielt eine Rolle, obschon E-Learning als wissenschaftliche Disziplin natürlich jünger ist. Allerdings ist die Nähe zur Wirtschaft, die an entsprechenden Forschungs- und Entwicklungsergebnissen interessiert ist, beim E-Learning natürlich größer. Anwendungsnutzen wird eher ökonomisch denn didaktisch taxiert.

¹L. Humbert hat auf der letzten INFOS diese Forschungslinien ausführlich nachgezeichnet. [Hu09]

Im Vergleich zu den anderen Schulfächern und ihren Didaktiken ist die Jugend ein entscheidender Faktor. Es gibt in der Didaktik der Informatik keine Forschungstraditionen sondern nur eine Vielzahl loser Fäden. Das Konzept der *fundamentalen Ideen der Informatik* [Sc93] ist ein solcher loser Faden ebenso wie die Arbeiten von Hubwieser [Hu00]. Lose Fäden miteinander zu verbinden scheint überdies nicht von übergeordneten Interesse. Eher wurden immer wieder neue Fäden aufgenommen, in dem z.B. die objektorientierte Modellierung und Programmierung aufgegriffen wurde, neue Technologien in Bezug auf ihren Nutzen untersucht wurden oder die Beziehung zur Medienbildung zumindest bildungspolitisch geklärt wurde [GI99].

3. DDI leistet zu viel bildungspolitische Arbeit:

Insbesondere die Klärung der Beziehung informatischer Bildung zur Medienbildung verweist auf einen Umstand der fachdidaktische Arbeit determiniert hat. Es sind in den letzten 15 Jahren eine Vielzahl bildungspolitischer Papiere entstanden, die zum einen auf die Notwendigkeit von informatischer Bildung für eine Allgemeinbildung in der sog. Informationsgesellschaft verweisen und zum anderen inhaltliche Schwerpunktsetzungen formulieren. Dadurch werden Ansprüche definiert, was informatische Bildung auf den verschiedenen Ebenen leisten soll. Problematisch hieran ist, dass eine Vielzahl von Ansprüchen darin genannt werden, zum Teil widersprüchlich sind und nicht mit der Realität (alltäglichen Praxis) des Informatikunterrichts abgeglichen wurden. Insbesondere der letzte Punkt soll im Folgenden am Beispiel der Bildungsstandards Informatik belegt werden. Dies geschieht nicht ohne einen Hinweis darauf, wo ähnliche Probleme auftreten.

3 Bildungsstandards der Informatik

Es ist allgemein bekannt, dass die Bildungsstandards der Informatik [GI08] nicht den selben Status haben wie z.B. die der Mathematik. Die Bildungsstandards der Informatik sind ein bildungspolitisches Papier der GI und damit Ausdruck von Lobbyarbeit. Diese ist zwar wichtig und an sich nicht zu kritisieren. Es ist jedoch problematisch, wenn zu viel versprochen wird. In der Einleitung der Bildungsstandards findet sich ein Abschnitt, für dessen Erfüllung noch kein Informatikunterricht und schon gar keiner, der in der Sekundarstufe I funktionieren würde, erfunden worden wäre. „In einer Zeit, in der Informatik immer mehr Lebensbereiche erfasst und Fachkräfte in der IT-Branche gesucht sind, brauchen Schülerinnen und Schüler zum einen fachliche Orientierung zur Einordnung der Informatik in ihrem persönlichen Umfeld, zum anderen müssen sie anschlussfähiges Wissen für eine vertiefte informatische Bildung und Ausbildung erwerben. Der Weg dazu liegt in frühzeitig erworbenen Kompetenzen im Fach Informatik.“ [ebd.]

Betrachtet man diese Sätze bildungstheoretisch, zeigen sich zwei grundsätzlich verschiedene Verständnisse, was Allgemeinbildung leisten soll.² Zum einen wird darauf abgezielt, gesellschaftliche Teilhabe zu ermöglichen (hier wären Emanzipation oder Aufklärung der Hintergrund, was auch Technikkritik mit beinhaltet); zum anderen wird die Re-

²Bildungstheoretische Konzepte zeichnen sich dadurch aus, dass sie normativ argumentiert sind. Auf der Grundlage früherer Veröffentlichungen wird vor dem Hintergrund eines Menschen- und Gesellschaftsbildes Forderungen aufgestellt, was Allgemeinbildung leisten soll. Unterschiedliche bildungstheoretische Entwürfe haben mindestens ein unterschiedliches Gesellschaftsverständnis.

krutierung von Nachwuchs im IT-Bereich beabsichtigt. Hier geht es eher um Werbung für spannende und auch gut bezahlte Berufe als Teil einer Spezialbildung, die von Allgemeinbildung unterschieden werden muss. Tatsächlich fehlt es an Nachwuchs, es fehlt aber auch an kritischer Auseinandersetzung mit der IT-Entwicklung. Beides miteinander zu vereinbaren ist schwierig. Wie schwierig dieses ist, zeigt sich seit fast 50 Jahren in der Diskussion um die technische Bildung, die für den Versuch zwei unterschiedliche bildungstheoretische Konzeptionen zu vereinbaren eine Blaupause darstellt. Sie zeigt sich aber auch in der fachdidaktischen Diskussion um die Naturwissenschaften. Bevor auf letzteres eingegangen wird, sollen Daten aus der Schulstatistik NRW die prekäre Situation der Informatik in der Schule aufzeigen. Dabei werden dann erste Ausblicke auf weitergehende Forschungen gegeben.

4 Daten zum Informatikunterricht in NRW und ihre Interpretation

Auch für die Sekundarstufe I weist die Schulstatistik NRW³ eine Menge Informatikunterricht aus. Dies betrifft nicht nur die Gymnasien und Gesamtschulen, die im Differenzierungsbereich (früher Jahrgang 9 und 10, heute an den Gymnasien 8 und 9) z.T. Wahlpflichtkurse anbieten sondern auch die Haupt- und Realschulen. Auffällig ist jedoch, dass die Zahl der ausgebildeten Informatiklehrer an den zuletzt genannten Schulen nicht zu der Anzahl der Kurse passt. Es steht zu vermuten, dass hier – wie dies für den Informatikunterricht nicht untypisch ist – sehr viel fachfremd unterrichtet wird. Überdies wird es so sein, dass die Bezeichnung des Unterrichts nicht zu den Inhalten passt. Wahrscheinlich verbergen sich hinter der Bezeichnung Informatik *de facto* Schulungen in Word, Excel oder Powerpoint, Zehn-Finger-Schreiben oder im Umgang mit Anwendungssystemen für Grafik, Musik oder Video, die allenfalls als anwendungsorientierte Informatikunterricht durchgehen, in der Regel sich aber auf Anwendungsschulungen beschränken. Erforscht ist dieses jedoch nicht. Im Zuge einer entsprechenden empirischen Forschung wäre es interessant herauszufinden, welche der Kompetenzen, die in Bildungsstandards genannt werden, tatsächlich erreicht werden. Umgekehrt ist es für die Informatik problematisch, dass eine solche Ausrichtung des Informatikunterrichts durchaus im gesellschaftlichen Interesse zu liegen scheint, da viele Außenstehende (und zum Teil auch Schülerinnen und Schüler) eine solche inhaltliche Ausrichtung erwarten. Aber auch diese Erwartungshaltung von Außen ist bislang nicht systematisch erhoben worden.

In der Sekundarstufe II, in der das Fach Informatik einen höheren Stellenwert haben könnte, ist die Situation nicht viel besser. Zwar gibt es an Gymnasien und Gesamtschulen einen nennenswerten Anteil an ausgebildeten Informatik-Lehrern, so dass das Fach Informatik inzwischen fast flächendeckend an diesen Schulen angeboten werden kann. Allerdings wird es nicht so oft angewählt. Die Anwahlen liegen auch in der Jahrgangsstufe 11 (die seit dem Schuljahr 2010/11 „Eingangsphase“ heißt, da es an Gymnasien der 10. Jahrgang ist und nur noch an Gesamtschulen der 11. Jahrgang) weit hinter denen der „unbeliebten“⁴ Naturwissenschaften Physik und Chemie. Hierfür gibt es sicher eine Reihe von Gründen, die z.T. in den Rahmenbedingungen begründet sind. Es gibt auch Indi-

³http://www.schulministerium.nrw.de/BP/Schulsystem/Statistik/2009_10/StatUebers.pdf

⁴Vergleiche hierzu Merzyn in [Me08].

zien dafür, dass dies nicht die ganze Wahrheit ist. In den Tabellen 1 bis 3 sind die Vergleichszahlen der Fächer aus Aufgabenbereich III (ohne Mathematik) aufgeführt. Anwahlen zu den Kursen in den Fächern in NRW im Schuljahr 2009/10.⁵

	11	12		13	
		GK	LK	GK	LK
Biologie	88,26%	52,95%	26,12%	52,52%	26,08%
Chemie	40,79%	24,17%	43,60%	23,62%	3,94%
Physik	36,73%	21,90%	64,20%	21,52%	6,53%
Informatik	18,83%	9,97%	0,43%	9,64%	0,36%
gesamt	184,61%	108,99%	37,32%	107,31%	36,92%

Tabelle 1: Schülerinnen und Schüler gesamt

	11	12		13	
		GK	LK	GK	LK
Biologie	90,56%	61,06%	34,52%	61,39%	27,06%
Chemie	35,57%	22,00%	3,26%	21,17%	2,94%
Physik	21,56%	13,75%	2,20%	13,33%	2,36%
Informatik	9,53%	4,44%	0,13%	4,14%	0,09%
gesamt	157,22%	101,24%	40,10%	100,03%	32,45%

Tabelle 2: Schülerinnen

	11	12		13	
		GK	LK	GK	LK
Biologie	85,08%	43,23%	16,06%	41,52%	24,87%
Chemie	48,00%	26,78%	5,67%	26,66%	5,18%
Physik	57,68%	31,67%	11,47%	31,68%	11,70%
Informatik	31,67%	16,59%	0,79%	16,47%	0,70%
gesamt	222,44%	118,28%	33,99%	116,34%	42,45%

Tabelle 3: Schüler

In der Zeile „gesamt“ sind die Werte aus den Zeilen darüber addiert. Hierin zeigt sich, dass die Schülerinnen und Schüler schon im 11. Jahrgang durchschnittlich keine zwei Kurse aus dem Bereich der naturwissenschaftlich technischen Fächer wählen. Bei den männlichen Schülern sind es ein bisschen mehr als zwei. Es sind allerdings nicht nur die im folgenden zunächst zu nennenden zweifelsohne vorhandenen schlechten Rahmenbedingungen, die diese Zahlen hervorrufen.⁶

⁵Schulstatistik NRW a.a.O. Angaben in Prozent, mit Schülerinnen und Schüler im Fach Mathematik = 100%. Die Angaben zu den Schülerinnen und Schülern in der Oberstufe durchaus leicht unterschiedlich sind. Diese Zahlen sind über die letzten fünf Jahre im übrigen nahezu konstant, so dass wir uns auf diese Momentaufnahme beschränken können.

⁶In diesem Abschnitt werden auf der Grundlage von Gesprächen mit Kollegen sowie Schülerinnen und Schü-

Zum einen werden durch ein Fach, das höchstens im Wahlpflichtbereich der Sekundarstufe I besteht, viele Schülerinnen und Schüler nicht erreicht, die sich dann in der Oberstufe nicht trauen, das Fach zu wählen. Fächer, die in der Unter- und Mittelstufe nicht vorkommen, haben es wohl schwerer in der Oberstufe. Für eine Belegung über die Eingangsphase hinaus (in der sog. „Qualifikationsphase“) ist das Fach Informatik nicht gleichberechtigt zu den Naturwissenschaften wählbar. Es kann nur als „zweite Naturwissenschaft“ von solchen Schülerinnen und Schülern gewählt werden, die ihre Fremdsprachenbelegung bereits erfüllt haben.

Im Vergleich mit Physik und Chemie sind die Zahlen aber so viel kleiner, dass die schlechten Rahmenbedingungen nicht der einzige Grund sein kann. Denn vor allem die hohe Abwahlquote von knapp 50% bei den männlichen und von über 50% bei den weiblichen Schülern erklärt diese nur bedingt. Denn insgesamt belegt immerhin noch die Hälfte aller Schülerinnen und Schüler eine zweite Naturwissenschaft in der Oberstufe, die allerdings schon bei der Wahl für die Eingangsphase vorbestimmt wird. Auffällig ist zudem die geringe Quote an Leistungskursen, die wohl auch mit der geringen Zahl der Lehrer zusammenhängt, aber auch durch die geringen Anwahlzahlen bedingt ist. Es gibt auch an Schulen, wo eine solche Wahl möglich ist, nur wenige Schüler und nach den Zahlen keine Schülerinnen, die das wollen würden. Oft müssen zwei Schulen zusammenarbeiten, um einen Leistungskurs einzurichten.

Die Quote an jungen Frauen, die in den Leistungs- und Grundkursen sitzen, ist überdies so gering, dass an der inhaltlichen Orientierung, an der Methodik oder gar an dem Medieneinsatz in diesem Fach etwas nicht stimmen kann. Die Inhalte dessen, was im Informatikunterricht gemacht wird, passen nicht zu den Vorstellungen vieler Schülerinnen und Schüler. Deren Sichtweise auf die Informatik ist sehr viel anwendungsorientierter als es die Lehrpläne und Vorgaben zum Zentralabitur vorsehen. Außerdem wird kritisiert, dass der Unterricht zu sehr auf das Programmieren am Computer ausgerichtet sei. Die Frage, wofür man diese Fähigkeiten brauche, wird gestellt und kann nicht genügend gut beantwortet werden. In der Tat ist das Programmieren eine hochspezialisierte Tätigkeit, die nicht jeder beherrschen muss und schon deswegen wenig allgemein bildend ist. Sie passt auch gar nicht zu dem oben bereits zitierten Begründungszusammenhang einer gesellschaftlichen Teilhabe, auf die durch Informatikunterricht vorbereitet werden soll. Sie passt allenfalls zu der Absicht wissenschaftlichen bzw. technischen Nachwuchses zu rekrutieren. In der Tat ist – dies würde einen weiteren Beitrag füllen – das Programmieren auch in seiner Einkleidung als Modellieren unter allgemeinbildenden Gesichtspunkten fragwürdig.

Das Programmieren ist auch ein nicht (mehr) sehr typischer Umgang mit Computern, da hier sehr viel mehr und sehr viel deutlicher strukturelles und erfindendes⁷ Denken gefordert und weniger gefördert wird. Die algorithmische und/oder objektorientierte Modellierung eines Problems ist jeweils nur eine spezifische Form des problemlösenden Denkens, die nicht von allen Schülern in gleicher Weise geleistet werden kann. Das Programmieren ist nicht nur eine hochspezialisierte sondern eine viele Schüler wenig moti-

lern Vermutungen formuliert, die zu Hypothesen weiterverarbeitet werden sollen.

⁷Viele Aufgaben in der Informatik bestehen nicht darin bekanntes zu reproduzieren sondern einen spezifischen Transfer zu erreichen.

vierende Tätigkeit. Was manche mit viel Spaß betreiben, sehen andere, nach vorläufigen Eindruck die große Mehrheit der Schüler, als sinnfreies Tun.⁸

In diesem Zusammenhang auf das Implementieren zu verzichten, ist im übrigen auch keine Lösung. Ohne die Absicht, das Modellerte auch zu implementieren, wird Modellieren sinnentleert. Implementiert man aber, erinnert die Anzahl der beim Programmieren auftauchenden Fehlermeldungen an die Nutzung von Anwendungsprogrammen vor über 20 Jahren. Es ist zum Teil demotivierend, was man an Rückmeldungen erhält. Die Motivation wird in der Art einer sich selbst erfüllenden Prophezeiung im Verlauf des Kurses immer schwächer. Nicht nur aufgrund der Schwächen der Arbeitsumgebungen ist das Programmieren eine sehr zeitaufwändige Tätigkeit. Interessant wäre es herauszufinden, wie viel Zeit im Informatikunterricht erfolgreiche Schüler mit dem Fach verbringen. Ist dieser Zeitaufwand nicht mit dem anderer Fächer kompatibel, d.h. sehr viel größer, wäre das ein weiterer Hinweis auf die tieferliegenden Probleme des Faches.

Einem Teil der aufgeworfenen Fragen gehen wir zur Zeit vor Ort nach, um durch diese Vorstudie in einer breit angelegten schon durch die dann große Zahl beteiligter Schulen eine gewisse Repräsentativität zu erhalten. Dabei stehen derzeit vor allem die Fragen nach dem Wahlverhalten und der Gründe dafür im Zentrum.⁹

Im folgenden Abschnitt wird durch Verweis auf Studien in den Naturwissenschaften gezeigt, wie man die alltägliche Praxis in den Blick bekommt. Dabei wird z.T. zwar das eigene Nest insofern beschmutzt, da man zeigt, dass man gemessen an den selbst gesteckten Zielen wenig erfolgreich ist.

5 Zur Situation in den Naturwissenschaften

Eine kritische Auseinandersetzung mit Methoden und Inhalten der Fächer findet in den Naturwissenschaften schon seit Jahrzehnten statt. Hier wird grundsätzliches, wie z.B. auch die Orientierung am wissenschaftlichen Tun in Frage gestellt. Seit einiger Zeit, genauer seit den „schockierenden“ Ergebnissen der PISA-Studie 2000, scheint sich diesbezüglich nun auch etwas zu tun. Die grundlegenden Probleme waren aber schon vorher bekannt, was auch im Folgenden exemplarisch belegt wird.

Mathematik, Physik und Chemie sind Fächer, die polarisieren.¹⁰ An diesen Fächern scheiden sich die Geister viel mehr als an anderen Fächern. Die Gründe hierfür werden von einzelne Lehrer auch mitgeliefert. Diese Fächer sind in ihren Augen „hart“ und die anderen sind „weich“ oder gar „Laberfächer“. Diese anmaßende Haltung berücksichtigt z.B. Begabungsunterschiede der Schülerinnen und Schüler nicht. Dabei räumen gar die-

⁸Darunter sind auch Schülerinnen und Schüler, die das Fach weiter belegen müssen, weil sie sich frühzeitig darauf festgelegt haben, ohne zu wissen, worauf sie sich einlassen aber zugleich in anderen Bereichen wie z.B. der Automatentheorie oder der Kryptologie, z.T. sogar im Bereich von Aufgaben zu Datenbanken gute Leistungen zeigen.

⁹Mit dem Ablauf des Schuljahrs 2010/11 werden diese Vorstudien abgeschlossen sein, so dass sie zur INFOS vorliegen sollten.

¹⁰Mit Informatik lässt sich die Reihe fortsetzen, obschon allein hierfür, wie oben geschildert, die empirischen Belege fehlen.

jenigen, die Physik, Chemie und Mathematik kritisch sehen, deren Wichtigkeit ein.

Es ist vor allem ein schlechtes Lernklima, das diesen Fächern attestiert wird. Dies ist u.a. Folge der Inhalte bzw. ihrer Vielzahl und der schnellen Abfolge, in der sie präsentiert werden. Daraus resultiert eine Methodik, bei der man auf verstärkten Frontalunterricht und dem Lesen von Lehrtexten setzt. Experimentieren, Explorieren und Üben kommen zu kurz [Me08]; und das in den Naturwissenschaften, die wie sonst kaum andere Wissenschaften den Geist der Aufklärung in sich tragen. Eine Folge ist, dass Schülerinnen und Schüler die Sätze und Gesetze der Naturwissenschaften wie dogmatische Lehrsätze lernen. Dies ist auch nicht neu, denn schon in den 1980er Jahren, ließ sich als Resultat des naturwissenschaftlichen Unterrichts u.a. feststellen: Die „Entmystifizierung der mittelalterlichen Vorstellung von der Welt [wird] in der Forderung nach *Wissenschaftsorientierung*... ersetzt durch einen neuen Mystizismus positiven Partikularwissens; Pestalozzi sprach von *Brockenwissen*.“ [BH87]

H. Faulstich-Wieland verwies schon damals auf eine Studie an Marburger Schulen. Naturwissenschaftlicher Unterricht führe in seiner bisherigen Form zu einer „unkritischen Wissenschaftsgläubigkeit“ [Fa86], die es damit auch erschwert, Technik rational einzuschätzen. Auch die häufig geäußerte Behauptung, dass naturwissenschaftlicher Unterricht einen wesentlichen Beitrag zum Wirklichkeitsverständnis und zu einem kritischen Bewusstsein gegenüber Wissenschaft und Technik beitrüge, wird in empirischen Studien zum naturwissenschaftlichen Unterricht relativiert. Es steht zu vermuten, „dass die schulischen Naturwissenschaften in ihrer traditionellen Form wesentlich für jene gefährliche Verbindung von Abwehr und Respekt, von Angst und Gläubigkeit (mit)verantwortlich sind, die den notwendigen emanzipativen Umgang von Individuum und Gesellschaft mit Wissenschaft und Technik so schwer macht.“ [Br85]

Es geht an dieser Stelle nicht darum, mit den Fingern auf die Naturwissenschaften zu zeigen bzw. ihren Bildungswert zu diskreditieren. Es geht aber darum, den Unterschied von bildungspolitischem Wollen und den Ergebnissen in der Praxis aufzuzeigen. Die Vermutung, dass früher alles besser gewesen sei, dürfte durch den Blick auf inzwischen um die 25 Jahre alte Studien auch widerlegt sein. In den Lehrplänen der 16 Bundesländer steht viel, was insofern nicht umgesetzt wird, da es zwar unterrichtet, aber nicht von allen (oder gar der Mehrzahl der) Schülerinnen und Schülern gelernt wird. Dies betrifft nicht nur einige wenige Schülerinnen und Schüler sondern offenbar viele. Dabei kann nicht davon gesprochen werden, dass die Befunde von damals überholt sind, Merzyn kann diese bis heute belegen [Me08]. Die schwachen Ergebnisse der PISA-Studien (die sich inzwischen in mittelpträgige gewandelt haben) sind ein weiterer Beleg.

Merzyn zieht daraus den Schluss, dass man differenzieren müsse, je nachdem, mit welcher Schülerklientel man es zu tun hat. Die Banalität dieses Satzes, zeigt wie tiefgreifend das Problem ist und wie wenig dies durch geänderte Lehrpläne (als Ausdruck bildungspolitischer Absichtserklärungen) geändert werden kann. Es ist wohl auch ein Mentalitätsproblem der Lehrenden, die dann doch eher Fachwissenschaftler, die zwar in ihren Kompetenzen denen in Wissenschaft und Forschung unterlegen sind, denn Pädagogen sind. Die Untersuchungen, die Berger hierzu vor gut 15 Jahren durchgeführt hat [Be01], sollten entsprechend akzentuiert noch einmal durchgeführt werden.

Dennoch aber lässt sich auch etwas über Lehrpläne oder Richtlinien steuern. Merzyn fordert konkret, dass Grundkurse in der gymnasialen Oberstufe nicht die leicht abgespeckte Variante zum Leistungskurs sein dürften. Die Inhalte müssen deutlicher beschnitten werden, müssen mehr aus der lebensweltlichen Perspektive der Schülerinnen und Schüler hergeleitet werden, es müssen methodisch andere Schwerpunkte gesetzt werden und auch weniger Formalismen und damit auf „harte“ Wissenschaft gesetzt werden [Me08].

Was für den Unterschied LK und GK wichtig ist, ist für die Unterscheidung Sek. I und Sek. II noch wichtiger, da in der SI naturwissenschaftlicher Unterricht eine pflicht- und damit keine Wahlpflichtveranstaltung ist. Die Leistungs- und auch – das zeigen die Studien, die Merzyn zusammengestellt hat, sehr deutlich – die Motivationsunterschiede sind erheblich. Diesen ist durch Orientierung an Praxis oder Kontext beizukommen. Hier und da müsste der Phänomenologie der Vorrang vor der Wissenschaft gegeben werden. Dieses ist bereits angelegt, muss aber weiter ausgearbeitet werden, z.B. in den „...im Kontext“-Projekten.

7 Fazit und Ausblick

Merzyns Forderung die Kurse inhaltlich und methodisch nach Sekundarstufe I, Grund- und Leistungskurs zu differenzieren, ist ein wichtiger Ansatzpunkt auch für die Weiterentwicklung der Informatik als Schulfach. Dazu kann in der Sekundarstufe I auf die „Informatik im Kontext“-Ansätze zurückgegriffen werden. Grundkurse sollten den Kontext ebenso berücksichtigen und sich weniger als bislang dem Implementieren (und Modellieren) widmen. Dies sollte vor allem den Schülerinnen und Schülern im Leistungskurs vorbehalten sein.¹¹

Es muss an dieser Stelle ebenso eingeräumt werden, dass nicht ganz klar ist, wie die hier gerade skizzierte Ausrichtung den Wünschen der Schülerinnen und Schülern und den von Außen an die Informatik herangetragenen Forderungen entsprechen. Daten hierzu werden derzeit ermittelt. Aus dem Spannungsfeld dieser Wünsche und den von den Fachdidaktikern vorgetragen Ansprüchen an Informatikunterricht ergibt sich ein pragmatischer Zugang zu dem, was informatische Bildung sein kann oder soll. Dies konnte auch Platzgründen hier nicht ausführlicher begründet werden und stellt im Zusammenhang der zu erhebenden Differenzen zwischen Wunsch und Wirklichkeit einen neuen Forschungsansatz zur Definition informatischer Bildung dar.

In diesem Beitrag sind hierzu eine Reihe von empirischen Forschungsansätzen genannt worden, mit denen der Status Quo des Faches in der unterrichtlichen Praxis erhoben werden soll, die vor allem die Gründe für die geringen Anwahlzahlen in den Blick nehmen. Eine gründliche Bestandsaufnahme dessen, was landauf und landab im Informatikunterricht getan wird, ist notwendig, um einen Ausweg aus den seit 20 Jahren erkannten Problemen des Informatikunterrichts zu finden. Zu den oben genannten und zugegebenermaßen noch auf Vermutungen (noch keine Hypothesen) beruhenden Fragen liegen keine durch Zahlen belegten Aussagen vor. So wissen wir nicht, was typisch oder gar reprä-

¹¹Aus Platzgründen mussten Überlegungen zum Projektunterricht zur Produktion von Software ausgelassen werden, die ebenso empirischen Untersuchungen zugeführt werden sollen.

sentativ ist. Wenn diese Zahlen vorlägen, könnten daran anknüpfend weitere Studien durchgeführt werden, in dem die Potenziale informatische Bildung verglichen mit den selbst gesteckten Ansprüchen erhoben werden könnten. Diese Daten und Fakten der alltäglichen Praxis erst einmal quantitativ zu erheben, ist ein erster Schritt.

Forschungen zur alltäglichen Praxis des Informatikunterrichts gibt es bereits. Diese z.B. von Rabel und Oldenburg [RO09] auf der letzten INFOS vorgelegten Ergebnisse betreffen allerdings mehr die Motivlage derer, die das Fach gewählt haben und fragen nicht so sehr nach den Gründen der Nicht- oder Abwahl. Auch eine sehr neue Untersuchung von Barthel, die als Dissertation am IPN in Kiel entstanden ist, zeigt Probleme und Potenziale des alltäglichen Informatikunterrichts auf [Ba10], auf die durch ein Review zu diesem Artikel unsere Aufmerksamkeit gelenkt wurde. Dem selben Review verdanken wir den Hinweis auf einen Umgang in der Mathematik-Didaktik mit dem alltäglichen Unterricht [KF10]. Die dort genutzten Forschungsmethodiken, die auf die Mikroebene des Unterrichts zielen, sollen zukünftig nach Abschluss der in diesem Artikel genannten Studien ebenso einbezogen werden.

Literaturverzeichnis

- [Ba10] Barthel, H.: Informatikunterricht. Wünsche und Erwartungen von Schülerinnen und Schülern. <http://www.mona-barthel.de/dissolendfassung.pdf> (gepr. 28.4.2011)
- [Be01] Berger, P.: Computer und Weltbild. habitualisierte Konzeptionen von der Welt der Computer. Westdeutscher Verlag. Wiesbaden. 2001
- [BH87] Bussmann, H.; Heymann, H.W.: Computer und Allgemeinbildung. In: Neue Sammlung 27 (1987) Heft 1, S. 2-39
- [Br85] Brämer, R.: Böse Erinnerungen. In: betrifft: erziehung 15 (1985), Heft 11, S. 48-52
- [Fa86] Faulstich-Wieland, H.: „Computerbildung“ als Allgemeinbildung für das 21. Jahrhundert? In: Zeitschrift für Pädagogik, 32.Jg. 86, Nr. 4, Beltz--Verlag, Weinheim, S.503-514
- [GI99] Gesellschaft für Informatik (Hrsg.): GI-Empfehlung: Informatische Bildung und Medienerziehung. LOG IN 19(1999)6
- [GI08] GI e.V.: Grundsätze und Standards für die Informatik in der Schule. LOGIN-Verlag, Berlin, Arbeitskreis „Bildungsstandards“ der GI. 2008.
- [Hu00] Hubwieser, P.: Didaktik der Informatik. Grundlagen, Konzepte, Beispiele. Springer. Berlin Heidelberg New York u. a., 2000
- [Hu09] Humbert, L.: Informatikdidaktik – Einschätzung der Landschaft. In: Peters, I.-R. (Hrsg.): Informatische Bildung in Theorie und Praxis – 25 Jahre »INFOS – Informatik und Schule«. INFOS 2009 – 13. GI-Fachtagung Informatik und Schule, 21.-24. September 2009 in Berlin. LOGIN-Verlag, Berlin, 2009.
- [KF10] Krummheuer, G.; Fetzter, M.: Der Alltag im Mathematikunterricht. Beobachten – Verstehen – Gestalten. Spektrum Akademischer Verlag. Heidelberg. Unv. Nachdruck, 2010
- [Ko09] Koerber, B. (Hrsg.): Zukunft braucht Herkunft. 25 Jahre »INFOS – Informatik und Schule«. INFOS 2009 – 13. GI-Fachtagung Informatik und Schule, 21.-24. September 2009 in Berlin. LNI Volume P-156 GI. Bonn. 2009
- [Me08] Merzyn, G.: Naturwissenschaften Mathematik Technik – immer unbeliebter? Schneider Verlag Hohengehren. Baltmannsweiler.
- [RO09] Rabel, M.; Oldenburg, R.: Konzepte, Modelle und Projekte im Informatikunterricht – Bewertungen und Erwartungen von Schülern und Studenten. In [KO09], S. 146 - 156
- [Sc93] Schwill, A.: Fundamentale Ideen der Informatik. In: Zentralblatt für Didaktik der Mathematik 25 Heft 1 (1993) S. 20-31.

Stereoskopische 3D-Videos selbst erstellen

Beat Trachsler^a, Martin Guggisberg^b, Martin Lehmann^c

^aKantonsschule Zürcher
Oberland
Bühlstrasse 36
CH-8620 Wetzikon
beat.trachsler@kzo.ch

^bDepartement Informatik
Universität Basel
Klingelbergstrasse 50
CH-4056 Basel
martin.guggisberg@unibas.ch

^cPHBern
Institut Sekundarstufe II
Muesmattstrasse 27a
CH-3012 Bern
martin.lehmann@phbern.ch

Abstract: 3D-Filme respektive stereoskopische 3D-Videos sind spätestens seit dem Kinoerfolg des “Fantasy-Films“ *Avatar* sehr populär. Die Thematik 3D-Film stößt bei Schülerinnen und Schülern auf großes Interesse. In diesem Artikel wird ein gangbarer Weg zur Erstellung eines computergenerierten 3D-Videos im Klassenzimmer beschrieben. Es werden die verschiedenen Stationen vom 3D-Modell der räumlichen Szene zum Webvideo im Internet aufgezeigt. Im Weiteren wird eine *low-cost* Lösung zur Präsentation eines 3D-Videos im Klassenzimmer auf der Basis der Polarisationsfiltertechnik vorgestellt.

1 Einleitung

Das Jahr 2010 war in vieler Hinsicht das Jahr des stereoskopischen Films. Nicht nur dass Hollywood nach dem gewaltigen Erfolg von *Avatar* im Herbst 2009 mit einer ganzen Reihe von stereoskopischen Produktionen nachzog. Auch die Internationale Funkausstellung Berlin (IFA 2010) stand unter dem Motto 3D. Wie sich diese Euphorie um den 3D-Film als Motivation für den Programmierunterricht nutzen lässt, wird in [MR10] aufgezeigt. Genau wie in jenem Artikel beschrieben, wird auch in diesem Beitrag der Raytracer POV-Ray¹ zur Bildsynthese eingesetzt. Dabei lässt sich die Theater-Metapher in der Schule gut ausreizen: Die Schülerinnen und Schüler planen die Positionierung von Kamera, Lichtquelle und den darzustellenden Objekten in ihrer Szene im Rahmen einer kleinen Projektarbeit. Ihren Plan setzen sie hinterher in der POV-Ray eigenen Programmiersprache um. Oft bestehen die darzustellenden Objekte aus mehreren Teilobjekten, welche mit Hilfe eines Algorithmus dargestellt werden können. Im nachfolgend beschriebenen Beispiel mit dem Menger-Schwamm wird beispielsweise ein rekursiv definiertes Makro eingesetzt. Damit die Schülerinnen und Schüler erfolgreich mit POV-Ray arbeiten können, müssen sie das von ihnen geplante 3D-Objekt formal erfassen und beschreiben können. In dieser Beschreibung sind neben der geometrischen Anordnung auch physikalische Parameter wie die Lichtintensität, die Durchlässigkeit von Medien oder die Art der Lichtquelle erforderlich. Im zweiten Teil dieses Artikels wird in Anlehnung an [ZF04] aufgezeigt, wie auch im Klassenzimmer mit vertretbarem Aufwand 3D-Videos erstellt und vorgeführt werden können.

¹ Siehe <http://www.povray.org>.

Im Rahmen eines Projektkurses zur 3D-Computergrafik werden an der Kantonsschule Zürcher Oberland, einem Schweizer Gymnasium, 3D-Videos erstellt und am Ende im Klassenzimmer vorgeführt. Der Projektkurs richtet sich in erster Linie an Gymnasiastinnen und Gymnasiasten mit Schwerpunktfächern Anwendungen der Mathematik und Physik im letzten Jahr vor der Maturprüfung², die bereits über ein Jahr Programmiererfahrung verfügen. In diesem Fall lässt sich das Werkzeug POV-Ray in drei Doppelaktionen einführen³. In einer weiteren Doppelaktion werden die Grundlagen der Stereoskopie vermittelt. Die bereits erwähnte Projektarbeit bildet schließlich den Höhepunkt des Kurses. Als Vorbereitung darauf erhalten die Schülerinnen und Schüler eine kurze Einführung ins agile Projektmanagement bei Softwareentwicklungsprozessen. Erfahrungsgemäß stecken sie sehr viel Zeit in die detailgetreue Ausarbeitung der Szene, wie das folgende Beispiel auf YouTube zeigt: <http://www.youtube.com/watch?v=hTl4yKFFhmg>. Vergleichbare Informatik Projektkurse⁴, bei welchen in Teams über einen längeren Zeitraum gearbeitet wird, wurden von Schülerinnen und Schülern ausschließlich positiv bewertet. Eine umfangreiche Sammlung von POV-Ray Projekten dieser Art ist auf <http://goodpractice.epistemis.com/> zu finden. Die dort beschriebenen Projektideen reichen vom virtuellen Vergnügungspark über die animierte DNA bis hin zum Schwarmverhalten von Zugvögeln. Aus Sicht der Lehrpersonen beobachten wir eine sehr hohe Motivation bei Schülerinnen und Schülern und einen hohen Einsatz auch über die Unterrichtszeiten hinaus. Das didaktische Potenzial liegt zentral in der direkten und raschen visuellen Verifikation der geplanten Realisierung der 3D-Szenen mit POV-Ray im Sinne eines explorativen Lernprozesses.

2 Vom POV-Ray Programm zum stereoskopischen 3D-Video

Ein stereoskopisches Einzelbild (engl. Frame, beim Video) besteht aus zwei Teilbildern, einem Bild für das linke Auge und einem Bild für das rechte Auge, welche im Gehirn des Betrachters zu einer räumlichen Wahrnehmung verschmelzen. Zur Erzeugung dieser Teilbilder werden zwei Kameraobjektive verwendet, wobei deren Abstand idealerweise gerade dem Augenabstand eines durchschnittlichen Betrachters (ca. 6.5 cm) entsprechen sollte. In diesem Abschnitt wird der Herstellungsprozess eines 3D-Videos im Detail beschrieben. Als erstes wird die Visualisierung mit dem Werkzeug POV-Ray diskutiert. Das zweite Unterkapitel beschreibt die Herstellung eines Webvideos. Im letzten Unterkapitel wird eine Möglichkeit zur Publikation des 3D-Videos im Internet vorgestellt.

Die Abbildung 1 zeigt ein Einzelbild aus einem 3D-Video. Bei dem dargestellten Objekt handelt es sich um einen Menger-Schwamm, ein dreidimensionales Fraktal. Im 3D-Video nähert sich die Kamera dem Fraktal und dringt schließlich ins Innere ein, wo sie auf einer gewundenen Bahn mehrere Ebenen des Fraktals passiert. Sämtliche Werkzeuge (Skripte und Programme) zur Erstellung dieses 3D-Videos sowie das 3D-Video selbst können auf http://goodpractice.epistemis.com/menger_schwamm.html bezogen werden.

² Schweiz. Abitur

³ Geeignete Tutorials zur Einführung in POV-Ray sind in [Hi03] und [Lo10] zu finden.

⁴ Informatik Studienwochen von Schweizer Jugend forscht, <http://fjb.informatik.unibas.ch/activities/index.html>

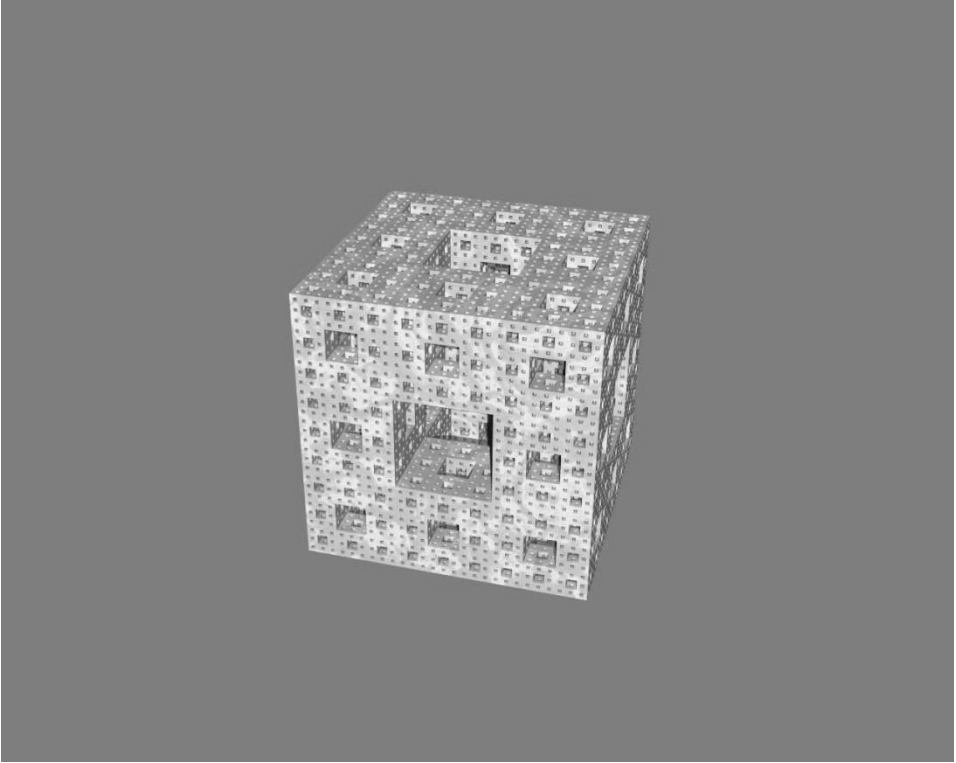


Abbildung 1: Menger-Schwamm mit Rekursionstiefe 4

2.1 Visualisierung des Menger-Schwamms mit POV-Ray

Ein Menger-Schwamm Fraktal kann wie folgt erzeugt werden: Als Ausgangspunkt dient ein Würfel. Jede Seitenfläche wird schachbrettartig in 9 gleich große Quadrate eingeteilt. Jeweils das mittlere Quadrat wird durchgehend von vorne nach hinten durchgestanzt. Es entsteht ein quaderförmiges Loch. Dieser Vorgang wird mit allen Seitenflächen des Würfels durchgeführt. Es verbleiben 20 Teilwürfel, die teilweise miteinander verbunden sind. Nun wird der Vorgang auf alle verbliebenen Teilwürfel angewendet. Mit Hilfe der Rekursion wird der Vorgang über mehrere Stufen wiederholt. In POV-Ray lässt sich diese Rekursion mithilfe von Constructive Solid Geometry (CSG) genau wie beschrieben realisieren.

Beim Raytracing mit POV-Ray werden ausgehend von der Kameraposition Sehstrahlen durch sämtliche Pixel der Bildebene geschickt. Diese Sehstrahlen werden rekursiv durch die virtuelle Szene verfolgt, wobei die Gesetze der Strahlenoptik (Reflexion, Brechung, etc.) berücksichtigt werden. Die Szene wird mit zwei virtuellen Objektiven gerendert, einmal für das linke und einmal für das rechte Auge. Die Einstellung der stereoskopischen Kamera erfordert zwei zusätzliche Parameter, den halben Augenabstand (Abstand der Objektive vom Mittelpunkt der Kamera) und den Abstand zum Fokuspunkt, auf den die beiden virtuellen Objektive fokussieren (Vgl. Abbildung 2). Dafür wird in Anlehnung an Arbeiten von Paul Bourke [Bo07] und Wolfgang Wieser [Wi04] ein Makro `setSimpleStereoCam` verwendet. Das Makro berechnet die Ausrichtung der Kamera derart, dass mit beiden Objektiven derselbe Ausschnitt der Szene fokussiert wird. Die folgenden Zeilen zeigen, wie das Makro eingesetzt wird:

```
//CAMERA
#declare EYE = 1; // 1 for right eye, -1 for left eye
#declare FL = vlength(LOOK_AT-CAMERA_POSITION); // focal length
#declare EYESEP = FL/30; // eye separation
camera {
    perspective
    setSimpleStereoCam(SKY, LOOK_AT, CAMERA_POSITION,
                      45, EYE * EYESEP / 2, FL)
}
```

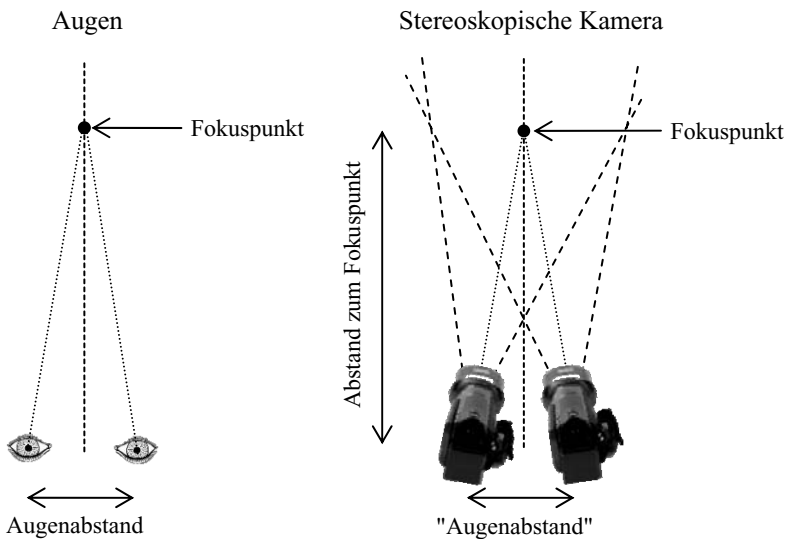


Abbildung 2: Stereoskopisches Sehen mit den Augen vs. stereoskopische Kamera

Mit der Variablen EYE lässt sich das Objektiv (linkes oder rechtes Auge) auswählen, mit dem die Szene gerendert werden soll. FL bezeichnet den Abstand zum Fokuspunkt LOOK_AT und CAMERA_POSITION steht für die Position der Kamera (genauer: für den Mittelpunkt zwischen dem linken und dem rechten Objektiv). Der Vektor SKY bestimmt die Ausrichtung der Kamera. Für den Augenabstand EYSEPP gilt die Faustregel, dass er gerade einen Dreißigstel des Abstandes zum Fokuspunkt betragen soll, damit die räumliche Wirkung der Szene optimal zu Geltung kommt. Dieser Wert lässt sich wie folgt motivieren: Schaut man im Abstand von 2 m durch ein Fenster nach draußen, so sieht man sowohl den Fensterrahmen als auch die Objekte draußen, vor dem Fenster scharf. Ein Dreißigstel von 2 m beträgt aber ungefähr 6,5 cm, entspricht also ziemlich genau dem durchschnittlichen Augenabstand beim Menschen.

Die Schülerinnen und Schüler sammeln bei ihrer Projektarbeit eigene Erfahrungen und probieren verschiedene Kameraeinstellungen selbst aus. Wie eine Schülerbefragung mit 50 Befragten gezeigt hat, schätzen es die meisten Schülerinnen und Schüler, wenn die Objekte vor der Leinwand erscheinen, weil dann der 3D-Effekt besonders stark zur Geltung kommt. Andererseits wird es in der Regel als unangenehm empfunden, wenn die virtuellen Objekte dem Auge zu nah kommen. Der Abstand der Objekte zur Kamera sollte daher immer mindestens so groß sein wie die Hälfte des Abstandes der Kamera zum Fokuspunkt. Weitere Tipps dieser Art sind beispielsweise in [Bo03] zu finden. Während der Projektarbeit verbessern und modifizieren die Schülerinnen und Schüler ihre Entwürfe kontinuierlich und motivieren sich dabei gegenseitig mit ansprechenden 3D-Visualisierungen. Bei der Animation von Objekten oder bei der Berechnung einer geeigneten Kamerafahrt setzen sie sich zudem mit der Parametrisierung von Bahnkurven im dreidimensionalen Raum auseinander.

2.2 Von der Bildsequenz zum Webvideo

POV-Ray liefert am Ende zwei Bildsequenzen, eine für das linke Auge und eine für das rechte Auge. Diese Bildsequenzen können nun beispielsweise mit einem Python Skript⁵ zu einer einzigen Bildsequenz von Megaframes der doppelten Breite zusammengefügt werden. In dieser neuen Bildsequenz sind die beiden Teilframes für das linke und das rechte Auge im Side-by-Side-Format nebeneinander abgelegt. Mit dem Add-on Firefog⁶ zum Firefox Browser 4 lässt sich daraus direkt im Webbrowser ein Webvideo im webM-Containerformat⁷ erzeugen. Wie das geht, wird im Tutorial auf unserer Webplattform ausführlich beschrieben. Alternativ kann das proprietäre Transkodierungsprogramm QuickTime Pro⁸ verwendet werden, um aus einer Bildsequenz ein Webvideo im mp4-Containerformat zu erstellen.

⁵ Ein passendes Python Skript findet sich auf http://goodpractice.epistemis.com/menger_schwamm.html.

⁶ Siehe <http://firefog.org>.

⁷ Siehe <http://www.webmproject.org>.

⁸ Siehe <http://www.apple.com/de/quicktime/extending/>.

2.3 Publikation des 3D-Videos auf YouTube

Ziel der Projektarbeit ist eine Veröffentlichung des 3D-Videos auf YouTube. Diese Plattform verfügt seit Mitte 2009 über einen integrierten 3D-Player zur Wiedergabe von stereoskopischen 3D-Videos. Dabei lässt sich im 3D-Menü des YouTube Players die passende Einstellung auswählen (Vgl. Abbildung 3). Das Video zum Menger-Schwamm ist zu finden unter <http://www.youtube.com/watch?v=SFcH6YcSp6c>.

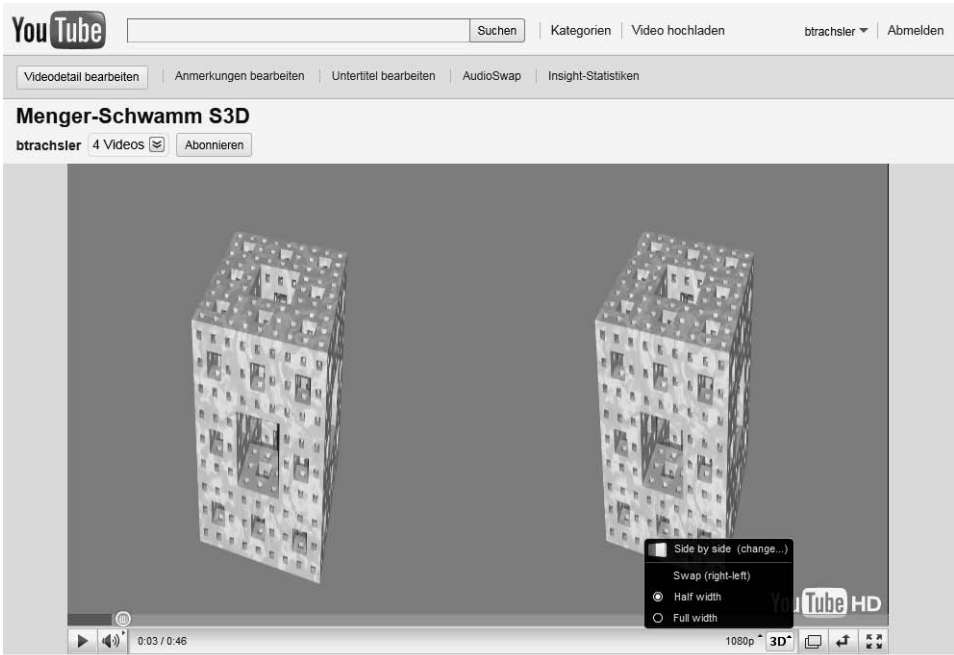


Abbildung 3: 3D-Video im YouTube 3D-Player

Zum Abspeichern des 3D-Videos auf YouTube wird ein YouTube oder Google Account benötigt. Das 3D-Video kann nun wie gewohnt hochgeladen werden. Bei den Einstellungen muss zusätzlich im Textfeld "Tags" das Tag `yt3d:enable=true` gesetzt sein, damit der 3D-Modus des YouTube Players aktiviert wird. Dabei ist zu beachten, dass der 3D-Modus von YouTube noch immer im Testbetrieb läuft. Eine detaillierte Dokumentation der `yt3d`-Tags existiert daher leider noch nicht. Im Folgenden eine kurze Zusammenfassung der wichtigsten Tags:

- `yt3d:enable=true` aktiviert den 3D-Modus des Webplayers.
- `yt3d:aspect=16:9` legt das Seitenverhältnis des Videos fest, in der Regel 16:9.
- `yt3d:swap=true` vertauscht das linke und das rechte Teilbild. Da der YouTube Player das rechte Teilbild in der linken Hälfte des Megaframes erwartet, ist die Standardeinstellung für unsere Videos „true“.

3 3D-Projektion im Klassenzimmer

Für die räumliche Wahrnehmung eines stereoskopischen Bildes ist entscheidend, dass jedes Auge nur das passende Teilbild und nicht auch noch Teile des anderen Teilbilds, sogenannte Geisterbilder, wahrnimmt. Dies wird mit einem 3D-Display erreicht. Für den Heimgebrauch werden mehrheitlich 3D-Displays mit Shutter-Brillen verwendet. Dabei tragen die Zuschauerinnen und Zuschauer Shutter-Brillen, welche abwechselnd ein Auge abdunkeln. Das Display sorgt dafür, dass zu jedem Zeitpunkt das passende Teilbild gezeigt wird. Damit kein Flimmern entsteht, sind Bildfrequenzen von über 100 Hz erforderlich. Daneben sind vor allem in den Kinos auch Displays im Umlauf, welche auf der Polarisations- oder Interferenzfiltertechnik beruhen. Dabei werden die Projektionsstrahlen für die beiden Teilbilder so modifiziert, dass sie mit einer zugehörigen Brille je nach Auge passend gefiltert werden können. Da sich Polarisationsfilterbrillen für wenig Geld erstehen lassen, konzentrieren wir uns auf diese Technologie. Dabei genügt es unserer Meinung nach, mit linearen Polarisationsfiltern zu arbeiten. Die zirkuläre Polarisationsfiltertechnik, die derzeit im Kino eingesetzt wird, hat zwar demgegenüber den Vorteil, dass man den Kopf während der Filmvorführung auch neigen kann. Dies spielt aber erst bei längeren Videosequenzen eine Rolle.

3.1 Materialliste

1. 2 identische Beamer
Die Beamer sollten in der Vertikalen justierbar sein, damit man die beiden Projektionen auf der Leinwand zur Deckung bringen kann. Zudem sollten sie über eine möglichst große Leuchtkraft verfügen, da die Polarisationsfilter die Lichtintensität auf unter 50% reduzieren.
2. Polarisationsfilterbrillen für linear polarisiertes Licht
Die beiden Filter für das linke und das rechte Auge sind dabei gegenüber der Horizontalen um 45° verdreht und weisen untereinander einen Winkel von 90° auf.
3. 2 lineare Polarisationsfilter
Die Filter werden vor den Objektiven der beiden Projektoren montiert. Dabei müssen sie genau wie die Brillen mit einem Winkel von 90° gegeneinander verdreht montiert werden, so dass die Polarisationsrichtung verschieden ist. Außerdem müssen die Polarisationsfilter auf die Polarisationsfilterbrillen abgestimmt werden, damit das Licht des linken Projektors auch tatsächlich das linke Auge erreicht.
4. Metallische Leinwand
Damit die Polarisation nicht verloren geht, braucht man eine metallische Leinwand. Die Leinwand kann auch selbst lackiert werden. Der Lack sollte in diesem Fall einen hohen Anteil Zink oder Aluminium enthalten.
5. Screen Splitter
Falls die Grafikkarte des PCs, mit dem man die Videos abspielen möchte, nur über einen zu den Beamern kompatiblen Ausgang verfügt, benötigt man zusätzlich einen Screen Splitter. Dieser kann in einem Fachgeschäft bezogen werden.

3.2 Installationsanleitung

Zunächst müssen die beiden Projektoren übereinander aufgebaut werden. Wir haben dieses Problem mit einer selbstgebauten Beamerhalterung aus Holz gelöst (Vgl. Abbildung 4). Die beiden Polarisationsfilter können dann mit einem Stativ vor die Objektive gespannt werden. Verbindet man nun die beiden Beamer beispielsweise über den Screen Splitter mit der Grafikkarte des PCs, ist die Hardware bereit. Auf der Softwareseite lassen sich die Videos im Prinzip direkt aus dem Browser abspielen. Falls bei der Einstellung der Grafikkarte oder des Screen Splitters Schwierigkeiten auftreten sollten, empfiehlt sich der Einsatz des Stereoscopic Players⁹. In diesem Tool lässt sich angeben, dass die stereoskopische Ausgabe auf zwei Displays erfolgen soll, was besonders für erste Tests sehr hilfreich ist. Damit die Videos im webM-Containerformat lokal abgespielt werden können, ist eventuell die Installation eines Plug-Ins¹⁰ erforderlich.



Abbildung 4: Beamerhalterung mit zwei DLP-Beamern und linearen Polarisationsfiltern

⁹ Siehe http://www.3dtv.at/Products/Player/Index_de.aspx.

¹⁰ Siehe <http://www.webmproject.org/tools/>.

4 Fazit

Das Paper stellt ein einfaches Verfahren zur Herstellung von computergenerierten 3D-Videos im Klassenzimmer vor und zeigt das didaktische Potenzial einer Projektarbeit mit 3D-Videos auf. Technisch lässt sich dies mit Makros für den Raytracer POV-Ray und geeigneten Tools zur Nachbearbeitung der Bildsequenzen an einer Schule problemlos umsetzen. Die Parametrisierung von bewegten Objekten im dreidimensionalen Raum ist nicht trivial und kann auf unterschiedlichen Niveaus mit Schülerinnen und Schülern erprobt werden. Dabei wird die räumliche Vorstellung wie auch die Modellierung physikalischer Eigenschaften von Lichtstrahlen geschult. Zudem ist die Thematik 3D-Film für die Schülerinnen und Schüler spannend. Mit der linearen Polarisationsfiltertechnik wurde ein Verfahren vorgestellt, mit dem sich solche 3D-Videos ohne große Kosten im Klassenzimmer vorführen lassen. Die stereoskopische Präsentationstechnik entwickelt sich derzeit sehr schnell und gewiss sind mancherorts bereits ähnliche 3D-Videoprojekte in Vorbereitung. Daher freuen wir uns auf neue 3D-Videos aus unseren Reihen und auf den Erfahrungsaustausch mit interessierten Kolleginnen und Kollegen.

Literaturverzeichnis

- [Bo03] Bourke, P: Creating stereoscopic images that are easy on the eyes. Februar 2003.
<http://local.wasp.uwa.edu.au/~pbourke/miscellaneous/stereographics/stereorender/>.
- [Bo07] Bourke, P: Additional cameras, mostly stereoscopic, for PovRay. Oktober 2007.
<http://local.wasp.uwa.edu.au/~pbourke/miscellaneous/stereographics/povcameras/>.
- [Hi03] Hidber, H-R: POV-Ray Tutorial. September 2003.
http://goodpractice.epistemis.com/media/pdf/povray_tutorial.pdf.
- [Lo10] Lohmüller, F. A: Eine Einführung in die Szenenbeschreibungssprache von POV-Ray. April 2010. http://www.f-lohmueller.de/pov_tut/basic/povkurs0.htm.
- [MR10] Marinescu, I. L.; Rick, D: Ein 3-D-Grafik-Projekt für viele. LOG IN Nr. 163/164, 2010.
- [Wi04] Wieser, W: Creating stereoscopic left-right image pairs with POV-Ray, 2004.
<http://www.triplespark.net/render/stereo/create.html>.
- [ZF04] Zelle, J. M.; Figura, Ch: Simple, Low-Cost Stereographics: VR for Everyone. Norfolk, Virginia, USA : ACM, 2004. SIGCSE.

Informatikgeschichte im Informatikunterricht – Konzepte und Materialien

Alexander Best

Westfälische Wilhelms-Universität Münster
Fachbereich Mathematik und Informatik (FB 10)
Institut für Didaktik der Mathematik und der Informatik
Arbeitsbereich Didaktik der Informatik
alexander.best@uni-muenster.de

Abstract: In diesem Beitrag werden Konzepte der Geschichtsdidaktik auf die Informatiklehre unter Berücksichtigung bereits etablierter Methoden und Lehr-Lernformen transferiert. Es werden Konzepte und Materialien behandelt, um Themen der Informatikgeschichte für Lernende transparent und strukturiert für den Unterricht aufarbeiten zu können. Dazu werden exemplarisch historische Materialien als Gegenstände für den Informatikunterricht aufgegriffen und ihre didaktischen Potentiale herausgestellt.

1 Hintergründe

Die Informatik als wissenschaftliche Disziplin an deutschen Hochschulen weist eine Bestandszeit von rund 40 Jahren auf. Wie kaum eine andere Wissenschaft hatte und hat sie einen immensen Einfluss auf technologische, soziale und politische Veränderungen unserer Gesellschaft. Um die Genese heutiger Konzepte, wie z.B. soziale Netzwerke oder mobile Anwendungen erkennen zu können, bildet Geschichtsbewusstsein [Si01] einen Schlüsselfaktor. Aktuelle Thematiken und Erscheinungsformen innerhalb der Informatik stellen durch die Einbettung in den historischen Kontext für Lernende einen Zugang zur Vergangenheit und Zukunft der Disziplin und ihrer Inhalte bereit. Untersuchungen wie etwa [UO10] zeigen, dass SchülerInnen neuen Technologien mehrheitlich unkritisch oder gleichgültig begegnen. Sie werden als feste Bestandteile der Gesellschaft aufgefasst. Über Herkunft, Auswirkungen oder zukünftige Einsatzmöglichkeiten solcher Technologien findet – zumindest uninitiiert – keine Reflektion statt. Diese und andere Überlegungen haben zur Folge, dass sich die Didaktik der Informatik in den letzten Jahren vermehrt darum bemüht neue Konzepte zur Integration von informatikgeschichtlichen Inhalten in die Lehre auszuarbeiten, sowohl an Schulen wie auch an Hochschulen. Während über die Potentiale (und auch Nachteile) dieser Bemühungen bereits Erkenntnisse dargestellt wurden, fehlt es noch an konkreten Konzepten sowie an exemplarischen Lehr- und Lernmaterialien. Einige sollen im vorliegenden Beitrag verdeutlicht und aufbereitet werden.

2 Konzepte

Immanent bei der Integration von geschichtsspezifischen Inhalten in den Schulunterricht sind Konzepte, um diese in bestehende Lehrabläufe einfließen lassen zu können. Die bislang im Informatikunterricht eingesetzten Konzepte reichen hierfür nicht aus, da sie nicht für die Vermittlung von historischen Inhalten ausgelegt sind. Die Geschichtsdidaktik bietet sich aufgrund langjähriger Forschungsarbeiten und Evaluationen auf diesem Gebiet zur Untersuchung eines möglichen Transfers von Methoden auf die Informatiklehre an. Nachfolgend werden einige ausgewählte thematische Strukturierungskonzepte vorgestellt und für die Informatiklehre diskutiert. In Abschnitt 2.2 werden ausgewählte Lehr-Lernformen betrachtet und deren Vorteile für den Informatikunterricht.

2.1 Thematische Strukturierung

Das genetisch-chronologische Verfahren: Neue Erkenntnisse, Ansätze und Strategien werden stets von den vorherigen beeinflusst und haben zugleich Auswirkungen auf zukünftige Entwicklung. Dieser Umstand kann didaktisch aufgegriffen werden, um für die Lernenden zuvor zusammenhanglose Ereignisse temporal und kausal in Beziehung zu setzen. Besonders die Interdependenz zwischen Notwendigkeit, Umsetzung und Auswirkung konkreter Entwicklungen kann im Zuge dieses Verfahrens ansprechend verdeutlicht werden. Als Beispiel könnten nach Eulenhöfer [Eu98] der Plankalkül der 1940er Jahre, die Befehlspläne der 1950er und 1960er Jahre sowie das heutige Verständnis von Programmiersprachen dienen. Auch die Beziehungen zwischen Leibniz Aufforderung *Calculemus!*¹ und späteren Bemühungen auf dem Gebiet der Formalisierung und Automatisierung, wie sie Humbert [Hu09] vorschlägt, könnten in diesem Unterrichtsverfahren behandelt werden. Die chronologische Vermittlung dieser Themen verdeutlicht ihre Genese. Ohne diesen Umstand können die zuvor genannten Begriffe zwar thematisch erfasst werden, bleiben jedoch für die Lernenden jeweils autarke Einzelentwicklungen. Auch der Blick in die zukünftige Entwicklung des heutigen Verständnisses von Programmiersprachen bliebe somit verschlossen. Insbesondere die geschichtlichen Hintergründe der sogenannten „Softwarekrise“ eignen sich dafür, Aussagen über die Anforderungen an zukünftige Programmiersprachen treffen zu können. Die Genese des *software engineering* vom *writing* über *building* hin zum *growing* verdeutlicht, wie Konzepte hinsichtlich neuer Entwicklungen überdacht und angepasst werden müssen (s.a. [Th09]). Ebenfalls die Geschichte(n) konkreter Programmiersprachen und deren Zielsetzungen sowie Verbreitung zu behandeln, würde es den SchülerInnen ermöglichen die heutige Programmiersprachenlandschaft besser zu verstehen und Abschätzungen bezüglich Anforderungen an die Sprachen von Morgen treffen zu können. Hierzu gehört etwa die Tatsache, dass es niemals eine einzelne Programmiersprache für alle Anforderungen geben kann. Während Sprachen wie Java, C oder C++ eine hohe wirtschaftliche und auch teilweise universitäre Durchdringung erfahren haben, eignen sich für didaktische

¹ Hierin werden bereits um 1680 Ansätze zur kalkülhaften Beschreibung von Sachverhalten zwecks Abgleich unterschiedlicher Auffassungen erkennbar [Hu09].

Zwecke hingegen ältere, skriptbasierte oder grafische Programmiersprachen.² Während in der Literatur (s.o.) zumeist klassische Beispiele zu finden sind, können auch Themengebiete mit hoher Aktualität und einem starken Bezug zum Lebensumfeld der SchülerInnen in diesem Verfahren aufgegriffen werden. So erfordert die rapide Verbreitung von mobilen Endgeräten (*mobile computing*), die dezentrale Datenverarbeitung (*cloud computing*) oder die grundlegende Veränderung von Mensch-Maschine-Interaktion (Tastatur/Maus → *Touch*oberflächen) eine Neubewertung bestehender Programmier- und Sicherheitskonzepte. Für diese Entwicklungen gibt es in der Vergangenheit bereits vereinzelt Indikatoren, welche mit den SchülerInnen aufgearbeitet werden können. Zugleich lässt sich für die Zukunft prognostizieren, dass etwa die klassischen *Personal Computer* durch mobile Geräte verdrängt werden. SchülerInnen können erkennen, dass aktuelle Neuerungen Momentaufnahmen einer Entwicklung darstellen, welche in der Vergangenheit eingesetzt hat und sich in der Zukunft weiter an gesellschaftliche Ansprüche/Notwendigkeiten annähern wird. Auf weitere spezifisch unterrichtspraktische Beispiele wird in Abschnitt 3 Bezug genommen. Während die Lernenden bei diesem Verfahren auf die historische Wahrnehmung und Auswirkung von Ereignissen sensibilisiert werden, kann jedoch kein hoher Grad an Fachwissen vermittelt werden. Hierfür bieten sich die nachfolgenden Verfahren an.

Die Fallanalyse: Beim genetisch-chronologischen Verfahren ist es an ausgesuchten Stellen sinnvoll, mit Lernenden ein konkretes historisches Ereignis detaillierter zu betrachten. Hierfür bietet sich die Fallanalyse an. Sie wird ebenfalls in anderen Disziplinen, wie etwa der Wirtschaftswissenschaft, Soziologie oder Publizistik eingesetzt. Auch bei der Verbrechensbekämpfung kommen Fallanalysen zum Einsatz um Täterprofile zu erstellen. In allen Fällen wurde deutlich, dass die Verlagerung der Sichtweise vom *Absstraktum* hin zum *Konkretum* das Verständnis für die Materie vereinfachen und erhöhen kann. Insbesondere bei zeitgeschichtlichen Themen mit hohem Aktualitätsbezug und gesellschaftlicher Präsenz bietet sich dieses Verfahren an, da sich persönliche/emotionale Einstellungen der SchülerInnen zu einem solchen Thema positiv auf den Unterricht auswirken können. So würde sich beispielsweise bei der Behandlung von Datenschutzaspekten die Fallanalyse anbieten, um Erfahrungen der SchülerInnen zu sammeln und auf Erlebnisse einzugehen. Insbesondere bezüglich sozialer Netzwerke oder Passwortsicherheit können SchülerInnen i.d.R. konkrete Beispiele in den Unterricht einbringen. Ebenfalls die Ausführungen Brunnsteins [Br92] zu Computer-Unfällen eignen sich gut, um Aspekte der Softwareverifikation und -validierung in Form der Fallanalyse zu behandeln. Dabei kann sehr tiefgründig vorgegangen werden, da keine anderen Ereignisse thematisch/chronologisch zum Behandelten in Beziehung gesetzt werden. Ziel ist die Vermittlung eines hohen Grades an fachlichem Hintergrundwissen an die Lernenden. Die Fallanalyse kann weiterhin für den Themeneinstieg genutzt werden. Die SchülerInnen haben somit im Verlauf der Unterrichtsreihe einen fachlichen Bezugspunkt, zu dem neue und nicht derart detailliert betrachtete Inhalte in Beziehung gesetzt werden können.

² Interessant in diesem Zusammenhang sind Statistiken, wie etwa der *TIOBE Programming Community Index*, welche aktuelle und historische Daten bezüglich der Verbreitung von Programmiersprachen veröffentlichen. Siehe http://www.tiobe.com/index.php/tiobe_index

Die Konstellationsanalyse: Besonders die Geschichte der Informatik zeigt, dass keinesfalls von einem einzelnen Entwicklungsstrang innerhalb der Wissenschaft ausgegangen werden kann. So lassen sich nach Thomas [Th05] teilweise vollkommen unabhängig existierende Entwicklungen nachweisen, deren Zielsetzungen nur marginal voneinander abweichen. Ein prägnantes Beispiel für diesen Umstand seien die epochenübergreifenden und weltweiten Bemühungen auf dem Gebiet der Automatisierung. Ein einzelner stringenter Entwicklungsstrang vom Abakus bis in die Ära der Mikroprozessoren existiere nicht. Die Konstellationsanalyse – ursprünglich aus dem Ressourcenmanagement stammend [Gü08] – ermöglicht es dem Lehrenden unterschiedliche Entwicklungsstränge innerhalb der Informatik für die Lernenden zueinander in Beziehung zu setzen. Unterschiede, Gemeinsamkeiten, Ursprünge und gegenseitige Abhängigkeiten können aufgezeigt und – meist in Form einer Grafik – festgehalten werden. Diese könnte für das Beispiel „Programmierparadigmen“ – wie es Schubert/Schwill in [SS04] vorschlagen – die verschiedenen Lösungsstrategien für ein konkretes Problem zueinander in Beziehung setzen. Während bei imperativen Programmiersprachen die Frage nach dem „Wie?“ im Vordergrund steht, ist bei deklarativen Sprachen die Frage nach dem „Was?“ entscheidend. Zusätzlich könnte eine solche Grafik die bevorzugten Einsatzgebiete für deklarative und imperative Sprachen beinhalten. Dies verdeutlicht den SchülerInnen, dass die Anforderungen an Sprachen und deren Strategien vom Einsatzgebiet abhängig sind. Auch die Entwicklungsgeschichten konkreter Sprachen können hierzu Aufschlüsse geben. So zeigt etwa die Geschichte der Programmiersprachen, dass neue Entwicklungen auf älteren Konzepten aufbauten, jedoch konkrete Schwächen oder Inkonsistenzen verbessern sollten. Ebenfalls die Tatsache, dass imperative Sprachen und deklarative Sprachen häufig wechselseitig Konzepte voneinander übernahmen und in ihr jeweiliges Paradigma eingliederten, ist zu beobachten. Durch eine angemessene Aufarbeitung in grafischer Form können viele dieser historischen Entwicklungen und Abhängigkeiten der Programmierparadigmen für die SchülerInnen ansprechend visualisiert werden. Ein weiteres denkbares Themengebiet im Zuge der Konstellationsanalyse wäre die Gegenüberstellung von klassischen Enzyklopädien und dem Wikipedia Projekt. Es könnten die historische Ausgangssituation (im Falle der klassischen Enzyklopädien z.T. jahrhundertalt), die beiderseitige Entwicklung inklusive der Gemeinsamkeiten, Unterschiede und Wechselwirkungen, die Zielsetzungen sowie die Bedeutung beider Formen für die Gesellschaft heute betrachtet werden. Eventuell könnten daraus folgernd sogar Abschätzungen hinsichtlich zukünftiger Entwicklungen herausgestellt werden.

*Das individualisierende Verfahren*³: Einzig der Mensch in seiner Rolle als Erschaffer von Geschichte ist Gegenstand dieses thematischen Strukturierungskonzeptes. Veränderungen innerhalb der Informatik oder das Aufkommen neuer Impulse sollen dem Lernenden anhand mehrerer Persönlichkeiten vermittelt werden. Wesentlich ist – genau wie bei den vorherigen Verfahren – die multidimensionale Sichtweise auf konkrete Ereignisse zu ermöglichen und zu schärfen. Die Lernenden sollen eine Beziehung zu den behandelten Personen aufbauen und Ereignisse der Informatikgeschichte somit personalisiert

³ Es lassen sich Parallelen zum Konzept des durch Thomas [Th05a] beschriebenen *history lifts* aus der Chemiedidaktik erkennen. Allerdings wird dieses Verfahren erst dann eingesetzt, „wenn der heuristische Weg der Erkenntnisgewinnung und der Problemlösung für den Schüler zu schwierig wird“. Das genetisch-chronologische Verfahren der Geschichtsdidaktik hingegen ist für den durchgängigen Unterrichtseinsatz ausgelegt.

nachvollziehen können. Sogenannte „versteckte Ereignisketten“ [Gü08], wie beispielsweise „... es kam zur Softwarekrise“ oder „... leitete die Ära der Mikroprozessoren ein“ sollen vermieden werden, da dies zur Anonymisierung von Geschichte führt. Das individualisierende Unterrichtsverfahren eignet sich zur Auseinandersetzung mit den Lebensgeschichten bedeutender Persönlichkeiten aus der Informatik und deren Errungenschaften. So bietet es sich bei der Behandlung von frühen Rechenmaschinen an, den Zugang über die Lebensgeschichte Konrad Zuses oder John von Neumanns zu wählen. Hierbei können den SchülerInnen die Hintergründe und Intentionen, welche zur Entstehung etwa der Z1 oder dem *Manchester Mark 1* führten, aufgezeigt werden. Die Frage nach dem „Warum?“ kann von den SchülerInnen über die Lebensgeschichten zuvor genannter Persönlichkeiten selbst erschlossen werden. Handelt es sich bei den Lernenden um jüngere SchülerInnen, so hat sich in der Geschichtsdidaktik auch der Zugang zu Themen über fiktive Charaktere bewährt. Auch in diesem Verfahren bietet es sich an auf Personen einzugehen, welche eine hohe gesellschaftliche Präsenz aufweisen. So können beispielsweise über die Person Bill Gates oder Steve Jobs die Entwicklung von Betriebssystemen und die Veränderungen innerhalb der Softwarebranche der letzten 30 Jahre exemplarisch an die SchülerInnen weitervermittelt werden. Zudem könnten Personen wie Ken Thompson, Dennis Ritchie oder Linus Torvalds – den SchülerInnen mehrheitlich nicht bekannt – vorgestellt werden und Konzepte des Unix bzw. Linux Betriebssystems zu den zuvor genannten Betriebssystemen in Beziehung gesetzt werden, um Unterschiede und wechselseitige Einflüsse herauszustellen. Hier würde sich also eine Symbiose zwischen dem individualisierendem Verfahren und der Konstellationsanalyse anbieten.

Zur Umsetzung der in Abschnitt 2.1 aufgeführten thematischen Strukturierungskonzepte werden angemessene Lehr-Lernformen benötigt. Auch hier können aus der Geschichtsdidaktik Konzepte entlehnt werden, um die Lernmethoden des Informatikunterrichts zu erweitern.

2.2 Lehr-Lernformen

Die Quellenarbeit: Die Arbeit mit historischen Quellen ist die am meisten eingesetzte Lehr-Lernform innerhalb des Geschichtsunterrichts. Sie zeichnet sich dadurch aus, dass durch die Auswahl der Quellen zwar eine didaktische Reduktion erfolgt, jedoch i.d.R. keine didaktische Rekonstruktion.⁴ SchülerInnen sollen Kernaussagen und Intentionen von Quellen selbstständig erarbeiten. Zusatzinformationen können dabei weitergereicht werden, um *Termini technici* zu erläutern oder aus der Quelle selbst nicht ersichtliches Hintergrundwissen zu liefern. Ein Beispiel hierfür in der Informatik wären chiffrierte schriftliche Überreste der ENIGMA oder Briefe bzw. Aufzeichnungen bedeutender Persönlichkeiten wie etwa Edsger W. Dijkstra, Alan Turing, John von Neumann oder Kurt Gödel. Es scheint für die Informatik analog zur Geschichtswissenschaft sinnvoll, eine Unterscheidung zwischen zwei Grundtypen von Quellen vorzunehmen: Die der Traditi-

⁴ Ausnahmen können auftreten, falls es sich um eine sehr umfangreiche Quelle handelt. Wird die Quelle in logische Abschnitte aufgebrochen, kann es in der Unterrichtsplanung sinnvoll sein die Abfolge dieser Abschnitte abzuändern. Auch bei der Arbeit mit sehr umfangreichen bildlichen oder filmischen Quellen ist eine didaktische Rekonstruktion denkbar.

on und die des Überrestes. Wurde vom Autor einer Quelle die Intention verfolgt, der Nachwelt eine bestimmte Sichtweise zu vermitteln, spricht man von einer Tradition. Bei der Auseinandersetzung mit dieser Art von Quellen ist eine umfassende Quellenkritik notwendig, da die historische Faktizität durch persönliche Sichtweisen beeinträchtigt wird. Handelt es sich bei Quellen hingegen um zufällige Überlieferungen, denen ein rein dokumentierender oder praktischer Zweck zukam, so wird von einer Tradition gesprochen. Es wird keinerlei persönliche Sichtweise übermittelt, sondern die faktische historische Vergangenheit. Jedoch ist das Arbeiten mit Traditionen weitaus anspruchsvoller, da vom Autor keine Aufbereitung erfolgt ist und dies somit ausschließlich von der Lehrperson – ausgenommen es handelt sich um einen bereits ausgearbeiteten Unterrichtsentwurf – erfolgen muss. Exzerpte aus Konrad Zuses Publikation „Der Computer – Mein Lebenswerk“ würden anhand zuvor genannter Kriterien als Tradition behandelt werden müssen. Ein Dokument, wie die Patentanmeldung Z23624 der Prozessorarithmetik der Zuse Z1 vom 21.12.1936 würde hingegen in die Kategorie der Überreste fallen.⁵ In der Informatiklehre bietet es sich auch an, neben der Quellenarbeit im klassischen Sinn, neuere Formen und Einsatzgebiete im Unterricht zu erproben. Mit der Versionierung innerhalb der Softwareentwicklung stehen beispielsweise detaillierte Protokolle von der Entstehung bis zur Gegenwart zur Verfügung. Die Versionierungsprotokolle stellen somit ein historisches Dokument dar. Ebenfalls historischer Quelltext fällt unter diese Kategorie. Bei der Betrachtung von Quelltext vor den 1990er Jahren fällt auf, dass i.d.R. auf Anmerkungen verzichtet wurde und der Aspekt Effizienz der Les- und Wartbarkeit übergeordnet wurde. Die Folge ist, dass Quelltext aus dieser Zeit extrem schwer semantisch erfasst oder gar verändert werden kann. Die Quellenarbeit beschränkt sich jedoch nicht ausschließlich auf schriftliche Quellen, sondern beinhaltet auch Bilder (zu denen auch Fotografien gezählt werden) sowie filmisches Material. Im Geschichtsunterricht wird hierfür i.d.R. das Panofsky'sche Interpretationsschema [Pa67] – z.T. in abgewandelter Form – angewendet: Es erfolgt zunächst eine Bildbeschreibung. In dieser Phase sollen die SchülerInnen lediglich beschreiben, was sie sehen. Ziel ist es alle wesentlichen Elemente des Bildes mit den SchülerInnen herauszuarbeiten, welche später analysiert und interpretiert werden sollen. Anschließend erfolgt die Bildanalyse. Hierbei sollen die Bedeutungen der zuvor herausgestellten Elemente erfasst werden. Auch sollen SchülerInnen eine Bedeutungshierarchie für die Bildelemente erarbeiten. Während einige Elemente für die abschließende Interpretation von grundlegender Bedeutung sind, fällt anderen Elementen hingegen eine geringere Aussagekraft zu. In der abschließenden Bildinterpretation sollen die Bedeutungen der wesentlichen Bildelemente zueinander in Beziehung gesetzt werden, um eine zentrale Kernaussage über die intentionelle Absicht des Autors treffen zu können.

Visualisierung: Bei Visualisierungen ist es möglich durch Beteiligung der SchülerInnen Unterrichtsergebnisse festzuhalten, aber auch Unterrichtsgegenstände zu entwickeln. Dies wird in der Fachdidaktik Informatik beispielsweise durch *Mindmaps* praktiziert. Bei der Behandlung von geschichtsspezifischen Inhalten im Informatikunterricht besteht jedoch die Notwendigkeit neben kausalen Bezügen auch den zeitlichen Kontext abzubilden. Eine Möglichkeit dies zu tun sind Zeitleisten. Allerdings ist deren Einsatz in einigen Situationen nicht sinnvoll, etwa wenn eine sehr umfangreiche Zeitspanne mit sehr vielen

⁵ <http://www.zib.de/zuse/Inhalt/Texte/Chrono/30er/Pdf/0177.pdf>

Ereignissen dargestellt werden soll und falls zwischen den dargestellten Ereignissen vielschichtige kausale Bezüge existieren. Um diese Nachteile auszugleichen, empfiehlt die Geschichtsdidaktik den Einsatz eines sogenannten Geschichtsfrieses. Dabei handelt es sich um eine Vereinigung (im mathematischen Verständnis) zwischen der Zeitleiste und dem Schaubild. Anders als im Falle der Zeitleiste werden nicht lediglich stichpunktartige Einzelereignisse festgehalten, sondern ganze Textpassagen, Bilder o.ä. Auch gibt es keinerlei Beschränkung der dargestellten Dimensionen und somit können zuvor dargestellte zeitliche Abschnitte erneut unter anderen Aspekten oder mit einem höheren Ereignissaufkommen dargestellt werden. Der Geschichtsfries kann somit verwendet werden, um Unterrichtsinhalte eines ausgedehnten Zeitraumes – sogar eines gesamten Schuljahres [Gü08] – festzuhalten bzw. zu entwickeln. Für einen Geschichtsfries, welcher die 1940er bis 1970er Jahre abdecken soll, könnten beispielsweise folgende Elemente integriert werden:

- Fotografien von Großrechnern im Kontrast zu Fotos der frühen Heimcomputer der 1970er Jahre
- Statistiken über die Transistorzahlen der CPUs (Stichwort *Moore's Law*), Herstellungskosten und Leistungsfähigkeit (FLOPS)
- Bedeutende Ereignisse für die Informatik innerhalb dieser Zeitspanne: Softwarekrise, ARPANet/Internet, proprietäre Software contra quellenoffener Code, etc.
- Programmiersprachen als Zeugnisse für historische Denkweisen

Das Schreiben: Dem Schreiben kommt in allen gesellschaftswissenschaftlichen Fächern eine zentrale Rolle zu. Wie bei kaum einer anderen Arbeitsform können SchülerInnen beim Schreiben nicht existierendes Wissen oder Halbwissen ihrerseits erkennen und somit Defizite selbstständig aufarbeiten. Weiterhin setzt das Schreiben über einen konkreten Unterrichtsgegenstand voraus, dass SchülerInnen ihr Wissen selbstständig strukturieren und so neue Zusammenhänge erkennen können. Besonders bei Siefkes [Si01] wird deutlich, dass das Schreiben auch in der universitären Lehre den Informatikstudenten große Probleme bereitet. So ergibt sich zumeist erst bei der Bachelor- und Masterarbeit die Konfrontation mit der Erstellung eines längeren Textes. In der schulischen Lehre kann das Schreiben von Texten zudem mit der Visualisierung verbunden werden, um Ausstellungen mit den SchülerInnen durchzuführen. Das Schreiben beschränkt sich also keineswegs auf die Einzelarbeit. Im Gegenteil bietet das gemeinsame Schreiben der SchülerInnen miteinander aber auch mit Einbindung des Lehrers Vorteile, wie etwa dem gesteigerten Motivationsfaktor, die Enthierarchisierung zwischen Lehrer und SchülerInnen und das Fördern von partnerschaftlicher und gruppenorientierter Arbeit. Beispiele sind Handbücher und Dokumentationen zu Projekten, aber auch Evaluationen oder Vorabanalysen. Gerade im Informatikunterricht besteht die Möglichkeit dem Schreiben (etwa von Aufsätzen im klassischen Sinn), fachspezifische und moderne Methoden hinzuzufügen. So stellen Blogs eine aktuelle und für SchülerInnen ansprechende Möglichkeit des Schreibens dar. In Blogeinträgen ist es mit geringem Aufwand möglich einen Verweis auf eine Quelle zu setzen. SchülerInnen können sich so mit dem Zitieren – sicherlich in einer stark vereinfachten Form – vertraut machen. Bezüglich modernen und fachspezifischen Methoden des Schreibens im Informatikunterricht gibt es sicherlich

noch eine Fülle weiterer zu untersuchender Varianten, wie etwa Podcasts, Webquests, etc.

Nachdem nun eine Auswahl an methodischen Konzepten zur Behandlung von geschichtsspezifischen Inhalten der Informatik vorgestellt wurde, sollen im folgenden Abschnitt einige Materialien für den Unterricht verdeutlicht werden. Die wichtigste Anforderung an diese Materialien ist die Möglichkeit historische Ereignisse und Entwicklungen mit heutigen Themen und aktuellen Bezügen der Informatik verbinden zu können. Materialien für den Unterricht aufzuarbeiten, welche dieser Prämisse entsprechen ist dabei keinesfalls trivial. So steht der Abakus für ein frühes Beispiel der Automatisierung und Formalisierung. Konkrete Bezüge zu aktuellen (kern)informatischen Themen und der Lebenssituation der SchülerInnen hingegen sind zunächst nicht offensichtlich.

3 Materialien

Soziale Netzwerke: Die Interaktion mit sozialen Netzwerken gestaltet sich extrem einfach und lässt SchülerInnen auf den ersten Blick vermuten, dass im Hintergrund von z.B. Facebook keine aufwendigen Algorithmen oder Datenstrukturen eingesetzt werden. Jedoch integrieren gerade soziale Netzwerke eine umfangreiche Palette an historischen Algorithmen und Datenstrukturen, um geringe Ladezeiten für Suchanfragen o.ä. zu ermöglichen. Werden Begriffe in Abfragen eingegeben, so erhält man eine sofortige Liste an möglichen Resultaten. Vergleicht man diese Nutzererfahrung nun mit Gegenbeispielen, so kann den SchülerInnen veranschaulicht werden, welchen Unterschied die Verwendung trivialer Algorithmen oder Datenstrukturen ausmachen kann. Ausgaben, wie „Bitte warten Sie bis Ihre Suchanfrage beendet wurde...“, werden von Benutzern heutzutage nicht länger toleriert. Viele Konzepte heutiger sozialer Netzwerke lassen sich bereits sehr früh erkennen. Das *Chatten* beispielsweise, wurde bereits in den ersten Unix-Varianten mit dem Programm *talk(1)* ermöglicht:

„The **talk** utility is a visual communication program which copies lines from your terminal to that of another user.“ – Auszug aus dem *talk(1) manual*

Mit dem Programm *mesg(1)* war es zudem möglich den Empfang von Nachrichten zu gestatten bzw. zu untersagen. Es wurden also bereits Aspekte zur Kontrolle der Privatsphäre berücksichtigt, welche in heutigen sozialen Netzwerken ein zentrales Thema darstellt. Ebenfalls die Nutzung von Mikroblogging-Diensten wie Twitter, wurde in Unix bereits mit dem Programm *finger(1)* und der Erstellung der Datei *~/plan* vorweggenommen. Ein weiterer Themenkomplex, welcher im Zuge von sozialen Netzwerken behandelt werden kann, ist die Datensicherheit. Die Geschichte der Datenverarbeitung ist von Ihren Anfängen bis in die heutige Zeit mit Fällen von Datendiebstahl und Datenmissbrauch gekennzeichnet. Soziale Netzwerke zeigen eine Vielzahl von Sicherheitsaspekten, welche aus Fehlern in der Vergangenheit diesbezüglich gezogen wurden. Aspekte wie SSL, die Hinterlegung von mehreren Email Adressen oder Identitätsverifikation sind nur einige Beispiele. Weiterhin gestattet es etwa Facebook jedem Benutzer, als

Entwickler aktiv zu werden. Dies könnte für SchülerInnen eine erste und didaktisch sinnvolle Einführung in das Programmieren sein. Der Aspekt der Objektorientierung wird von Anfang an fokussiert. Zudem ist es gleich zu Beginn und mit geringen Programmierkenntnissen möglich, grafische Applikationen zu schreiben. In Hochsprachen, wie etwa C oder Java, erfordert dies bereits professionelle Programmierkenntnisse. Soziale Netzwerke erscheinen somit als ein ergiebiger Pool an geschichtsspezifischen Unterrichtsmaterialien für den Informatikunterricht.

Computerspiele: Ein weiteres Beispiel für eine umfangreiche Materialsammlung zu informatikgeschichtlichen Unterrichtsinhalten bieten Computerspiele. Werden ältere Spiele aus den 1970er Jahren betrachtet, so können Algorithmen wie etwa Wegfindungsroutinen oder Algorithmen zur Generierung von zufälligen Ereignissen direkt durch die Betrachtung des Spielablaufes herausgearbeitet werden. Auch bezüglich der Hardware sind Computerspiele ein geschichtliches Zeugnis. So nutzten sie (und tun es noch) sehr stark spezifische Aspekte einer Rechnerarchitektur, um die zur Verfügung stehende Rechnerkapazität optimal zu nutzen. Nur wenige Computerspiele wurden mit dem Ziel der Portabilität entwickelt. Dies unterscheidet ältere Spiele von den aktuellen, welche auf einer Vielzahl von Plattformen (PC, Playstation 3, XBOX 360, etc.) ausführbar sein müssen. Zudem sind Computerspiele Zeugnisse der Softwareindustrie im Allgemeinen und spiegeln die jeweiligen Probleme aber auch Vorteile einer bestimmten Dekade wieder. Während Computerspiele in den 1970er und 1980er Jahren beispielsweise von Hobbyisten in einem Team von wenigen Personen entwickelt und vermarktet wurden, stellt die Entwicklung heutiger Computerspiele ein Projekt mit hunderten von Mitarbeitern und Produktionskosten im zweistelligen Millionenbereich dar. Das Computerspielmuseum in Berlin bietet sich zudem an, um im Rahmen eines Ausflugs die Geschichte der Computerspiele enaktiv und auch interaktiv erleben zu können. Auch lassen sich im Bereich des Computerspiele-Journalismus sehr viel stärker und früher als anderswo die Auswirkungen der Verlagerung von Print Medien auf digitale Medien beobachten. Die zukünftigen Veränderungen in Bezug auf das Lesen, können hier bereits mit den SchülerInnen abgeschätzt und in einen historischen Kontext eingeordnet werden.

4 Ausblick

Zur Behandlung von geschichtsspezifischen Inhalten im Informatikunterricht werden neue Konzepte benötigt, da die bislang existierenden hierfür nicht ausreichend sind. Jedoch lassen sich diese Konzepte teilweise aus den Didaktiken anderer wissenschaftlicher Disziplinen transferieren. Ein Beispiel hierfür sind die Didaktiken der Naturwissenschaften. Insbesondere bei Thomas [Th05] wird deutlich, dass im Chemieunterricht existierende Konzepte zur Behandlung von geschichtsspezifischen Inhalten auch auf den Informatikunterricht übertragen werden könnten. Auch die in diesem Beitrag vorgestellten Konzepte aus der Geschichtsdidaktik scheinen sinnvoll und zur didaktischen Verbesserung des Informatikunterrichts in Bezug auf geschichtsspezifische Lerninhalte beitragend. Konkrete Aussagen über die Vorteile/Einsatzmöglichkeiten von Konzepten anderer Disziplinen bei der Behandlung von Informatikgeschichte können jedoch nur im Kontext unterrichtspraktischer Erhebungen gemacht werden. Welche Konzepte und Methoden sich im Informatikunterricht bewähren, bleibt bis dahin abzuwarten. Eine

weitere Voraussetzung hierfür ist die Ausarbeitung eines Clusters an Unterrichtsmaterialien für die Informatikgeschichte. Diese Aufgabe ist keinesfalls trivial, da Materialien einer Vielzahl an Ansprüchen genügen müssen. Dazu zählen Faktoren wie etwa Enaktivität, ausreichende Quellenlage schriftlichen/visuellen Charakters, Möglichkeiten der didaktischen Reduktion/Rekonstruktion, Einbettung in einen historischen Kontext sowie die Verdeutlichung von Abhängigkeiten mit anderen historischen Ereignissen der Informatik und die Verknüpfung von historischen Materialien und Persönlichkeiten.

Literaturverzeichnis

- [Br92] Brunnstein, K.: Computer-Unfälle. LOG IN, 12. Jg. (1992), Heft 3, S. 17–23.
- [Eu98] Eulenhöfer, P.: Disziplinengeschichte und die Disziplinierung der Geschichte. FfF Kommunikation, 15. Jg. (1998), Heft 2, S. 29–33.
- [Fo10] Fothe, M.: Kunterbunte Schulinformatik. Ideen für einen kompetenzorientierten Unterricht in den Sekundarstufen I und II. Berlin, 2010.
- [Gü08] Günther-Arndt, H.: Geschichtsmethodik. Handbuch für die Sekundarstufe I und II. 2. Auflage, Berlin, 2008.
- [Hu09] Humbert, L.: Ideengeschichte oder Archäologie. Geschichte der Informatik – das Unsichtbare ist der Kern. LOG IN, 29. Jg. (2009), Nr. 157/158, S. 20–24.
- [Pa67] Panofsky, E.: *Studies in Iconology. Humanistic Themes In the Art of the Renaissance*. Torchbook Ed., New York, 1967.
- [SS04] Schubert, S.; Schwill, A.: Didaktik der Informatik. Heidelberg u.a., 2004.
- [Sc09] Schwill, A.: Unterrichtshilfen in Informatik. LOG IN, 29. Jg. (2009), Nr. 160/161, S. 14–33.
- [Si01] Siefkes, D.: Schreiben und Geschichte als Zugang zur Informatik. FfF Kommunikation, 18. Jg. (2001), Heft 4, S. 11–13.
- [Th05] Thomas, M.: Vom Abakus bis Zuse. In: Friedrich, S. (Hrsg.): Unterrichtskonzepte für informatische Bildung. INFOS 2005; 11. GI-Fachtagung Informatik und Schule, 28.–30. September 2005 an der TU Dresden. Bonn, 2005, S. 185–196.
- [Th09] Thomas, M.: Skript zur Veranstaltung „Fachdidaktik Informatik – Gestaltung von Lehre und Unterricht“, SS 2009, Abschnitt 3.4.6 Software.
- [UO10] Unterbruner, U.; Otrell-Cass, K.: Wie sich Jugendliche Technik und neue Medien in einer Welt in 20 Jahren vorstellen: Ergebnisse der Studie Jugend-Zukunft-2008/09 mit Jugendlichen aus Österreich, Deutschland und Neuseeland. In: Zumbach, J.; Maresch, G. (Hrsg.): Aktuelle Entwicklungen in der Didaktik der Naturwissenschaften: Ansätze aus der Biologie und Informatik, Innsbruck u.a., 2010, S. 37–53.
- [We91] Weinberg, J.: Didaktische Reduktion und Konstruktion. In: Tietgens, H. (Hrsg.): Didaktische Dimension der Erwachsenenbildung, 1991, S. 130–150.

Übertragbarkeit singulärer MINT-Interesse-initiierender außerschulischer Maßnahmen

Thiemo Leonhardt, Philipp Brauner, Jochen Siebert, Ulrik Schroeder

Abteilung
RWTH Aachen
Ahornstraße 55
52074 Aachen

{leonhardt, brauner, siebert, schroeder}@cs.rwth-aachen.de

Abstract: In diesem Artikel wird die Auswertung einer singulären Informatik- und Technikinteresse-initiierenden Maßnahme geschildert. Dabei werden als maßgebliche Einflussfaktoren das Selbstkonzept, die Selbsteinschätzung, die Expertise im Umgang mit Technik und die Zukunftsperspektive im Informatik und naturwissenschaftlichen Bereich untersucht. Die Ergebnisse zeigen weder einen alters- noch einen geschlechtsspezifischen Effekt in der Wirkung des Kurses, jedoch erwartungsgemäß einen signifikanten Unterschied in der geschlechtsbedingten Prädisposition. Zur Untersuchung, ob erfolgreiche außerschulische monoedukative Maßnahmen auch in einem koedukativen Klassenverband gleiche Ergebnisse erzielen können, haben wir im zweiten Schritt eine kontrollierte Studie mit Schulklassen der Region Aachen durchgeführt. Durch diese Zufallsstichprobe können Alterseffekte oder ein Effekt durch die freiwillige Teilnahme am Kurs minimiert werden. Die Ergebnisse zeigen, dass erfolgreiche monoedukative Workshop- Konzepte auch in einem koedukativen Klassenverband vergleichbar wirken.

1 Motivation

Das Interesse junger Menschen an Technologie und MINT-Fächern zu fördern ist das Ziel zahlreicher Initiativen¹. Viele dieser Maßnahmen finden singulär in Form von ein- bis siebentägigen Workshops statt. Der Girlsday, Studieninformationstage und Sommercamps sind prominente Vertreter. Eine adäquate personelle Begleitung der Workshops vor Ort an den Universitäten und Hochschulen ist ein häufig auftretendes Problem der Umsetzung zahlreicher Initiativen. Eine planvolle Organisation und die Vor- und Nachbereitung der Kurse verursachen einen erheblichen Zeitaufwand, so dass eine Überprüfung der Effektivität der Maßnahmen nicht nur bildungspolitisch sondern auch wirtschaftlich sinnvoll ist.

¹ Über 160 verschiedene deutsche MINT-Initiativen in Deutschland. Vgl. <http://www.mintzukunft.de/>

Neben der Fragestellung wie eine singuläre Maßnahme konzipiert werden kann [LS09, HS05, SS05], so dass diese einen positiven Einfluss auf die Teilnehmenden insgesamt hat, ist es sinnvoll, die entscheidenden Voraussetzungen der Teilnehmenden für eine erfolgreiche Teilnahme zu analysieren.

Das Selbstvertrauen einer Person im Umgang mit Computern und das Selbstvertrauen, technische Sachverhalte zu verstehen, lässt sich mit der Selbstwirksamkeitserwartung im Bereich Computer beschreiben. Effektiv lässt sich mit dieser Variablen sowohl die Selbsteinschätzung der eigenen informatischen Fähigkeiten als auch der aufgeschlossene Umgang mit Technik beschreiben. Studien zeigen, dass hohe Selbstwirksamkeitserwartungen im Bereich Computer mit hoher Technikkompetenz und Technikakzeptanz korreliert [AZ07, LG03, BK98, BU95].

Deutschland erreicht bei PISA 2009 im Bereich Naturwissenschaften zwar einen Wert, der über dem Durchschnitt der OECD-Länder liegt, doch die Spitzengruppe bilden Shanghai (China), Finnland, Hongkong (China) und Singapur. Um den Anschluss an die Spitzengruppe nicht zu verlieren, besser noch um aufzuschließen, ist eine weitere Verbesserung der Ausbildung essentiell.

Aus diesem Grund ist neben der Frage der Effektivität einer Maßnahme die Übertragbarkeit auf einen Klassenverband interessant, da außerschulische Maßnahmen für spezielle Gruppen und meist monoedukativ konzipiert werden. Eine Übertragbarkeit in die Schule und damit in koedukative Gruppen wird in der Forschung als schwierig gesehen und sogar eine zeitweise Trennung der Mädchen und Jungen empfohlen [FH96, WE99].

2. Auswertung go4IT! Roboterworkshops

Als zu untersuchende singuläre Maßnahme für die Initiierung des MINT-Interesses wurde auf die bereits erprobten und vom Lehr- und Forschungsgebiet Informatik 9 an der RTWH Aachen regelmäßig durchgeführten go4IT!-Roboterworkshops zurückgegriffen [LS09].

Didaktisch orientieren sich diese Workshops an dem wissenschaftlich evaluierten und bewährtem Roberta-Projekt² des Fraunhofer IAIS (Institut für Autonome Intelligente Systeme), bei dem die Faszination von Robotern genutzt wird, um Schülerinnen im Alter von 12 bis 18 Jahren Naturwissenschaften, Technik und Informatik praxisnah zu vermitteln. In der Evaluation des Roberta-Projekts wurde nachgewiesen, dass Teilnehmerinnen nach Abschluss des Workshops häufiger ein MINT-Studium in Erwägung ziehen [PET07]. Ebenfalls konnten wir empirisch nachweisen, dass innerhalb eines Workshop die Programmierung greifbarer Gegenstände, wie z.B. einem Roboter, gegenüber der Computerprogrammierung vorzuziehen ist, da diese den Lernerfolg und die Einstellung gegenüber der Informatik der Teilnehmenden steigert [BL10].

² Vgl. <http://www.iais.fraunhofer.de/roberta.html> (31.01.2011).

Während das Roberta Konzept in regionalen Zentren den Verleih von Roboterbaukästen und die Weiterbildung von Kursleitern verfolgt und dies seit 2005 mit *Roberta-Goes-EU*³ auf die europäischen Nachbarländer ausweitet, verfolgen wir das Konzept, Schulen Roboter-Workshops als kostenlosen Service anzubieten. Damit erreichen wir nicht nur die ohnehin für die Thematik aufgeschlossenen Mädchen. Interessierten Schulen der Region werden zweitägige Workshops in der 6. und 7. Jahrgangsstufe für jeweils 12 - 14 Mädchen angeboten.⁴

Es ist darauf hinzuweisen, dass die Teilnehmenden den Workshop nicht als Teil des Unterrichts ansehen, da sie aus dem Klassenverband heraus genommen werden und in dieser Zeit nicht am Unterricht teilnehmen können. Zusätzlich werden die Workshops nicht von bekannten Lehrern gehalten, sondern von Studenten, die nur als Hilfesteller auftreten und nicht die Lehrerfunktion übernehmen. Aus diesem Grund werden die Workshop des go4IT! – Projektes als außerschulische Maßnahmen wahrgenommen.

Die folgenden Ergebnisse beziehen sich auf den Zeitraum Januar 2009 bis September 2010. Insgesamt haben 694 Kinder an den go4it! Workshops in diesem Zeitraum teilgenommen. Die meisten Teilnehmer und Teilnehmerinnen kommen aus unserer favorisierten Zielgruppe der Jahrgangsstufen 6 und 7 und sind zum überwiegenden Teil Mädchen. In seltenen Fällen werden jedoch die Kurse mit interessierten Schülerinnen und Schülern aus anderen Jahrgangsstufen von den Schulen aufgefüllt (vgl. Tabelle.2.1).

		Alter						
		10	11	12	13	14	15	Gesamt
Geschlecht	Männlich	1	22	34	12	2	18	89
	Weiblich	22	214	286	57	8	4	591
Gesamt		23	236	320	69	10	22	680

Tabelle 2.1

91,9 % der Teilnehmenden stammen aus der anvisierten Zielgruppe der Jahrgangsstufe 6 und 7. Von den 694 Teilnehmern sind 86,9 % weiblich, da go4it! in erster Linie an Mädchen gerichtet ist und hier ein größeres Potential in Initiierung von Technikinteresse vorliegt als bei den Jungen. Trotzdem werden auch koedukative Workshops sowie Workshops für Jungen durchgeführt und evaluiert.

³ <http://www.iais.fraunhofer.de/roberta-eu.html> (31.01.2011).

⁴ Wir folgen hier den Empfehlungen aus dem Abschlussbericht der Roberta-Begleitforschung [SCH05].

Zur Untersuchung der Wirksamkeit des Kurses wurde ein Pre-Post-Test Design gewählt. Es wird die Technikexpertise⁵, das Selbstkonzept⁶ in Bezug auf Informatik (Ich kann Informatik, wenn ich will), die Selbsteinschätzung⁷ im Umgang mit MINT Inhalten (Ich kann Informatik) und der Frage nach einer möglichen Zukunftsperspektive⁸ im MINT – Bereich (Schule, Studium, Beruf) erhoben. Dabei sind alle Skalen positiv skaliert, so bedeutet ein höherer Wert auch immer eine positive Zustimmung.

Einstellungen der Versuchsgruppe

Zwischen den 11- und 12-jährigen ist kein signifikanter Unterschied in Bezug zur Technikexpertise, dem Selbstkonzept und die Zukunftsperspektive festzustellen. Dagegen tritt bei der Selbsteinschätzung ein signifikanter Unterschied zwischen den Gruppen auf (Mittlerer Rang 11-jährige 262, 12jährige 289, $p < .05$, $Z = -2.0$). Die 12-jährigen zeigen dabei im Mittel eine positivere Selbsteinschätzung.

Wertet man Unterschiede zwischen den Mädchen und Jungen aus, so ergeben sich in den vier Skalen signifikante Unterschiede (vgl. Tabelle 2.3).

Ränge				
	geschlecht	N	Mittlerer Rang	Rangsumme
Selbstkonzept	männlich	89	408.17	36327.50
	weiblich	589	329.12	193853.50
Technikexpertise	männlich	89	464.77	41364.50
	weiblich	591	321.79	190175.50
Einschätzung	männlich	89	401.92	35771.00
	weiblich	587	328.88	193055.00
Zukunftsperspektive	männlich	89	462.96	41203.50
	weiblich	582	316.59	184252.50

Tabelle 2.2

Der mittlere Rang der Jungen ist auf den vier Skalen größer als der mittlere Rang der Mädchen, d.h. die positiven Ausprägungen auf den Skalen Selbstkonzept, Technikexpertise, Selbsteinschätzung und Zukunftsperspektive der Jungen ist signifikant größer.

Statistik für Test⁹

⁵ 8-Item Skala, Wertebereich: 1 – 4.

⁶ 4-Item Skala, Wertebereich: 1 – 4.

⁷ 4-Item Skala, Wertebereich: 1 – 4

⁸ 4-Item Skala, Wertebereich: 1 – 4

⁹ Gruppenvariable: geschlecht

	Selbst- konzept	Technik- expertise	Ein- schätzung	Zukunfts- perspektive
Mann-Whitney-U	20098.500	15239.500	20477.000	14599.500
Wilcoxon-W	193853.500	190175.50	193055.000	184252.500
Z	-3.580	-6.414	-3.325	-6.685
Asymptotische Signifikanz (2-seitig)	.000	.000	.001	.000

Tabelle 2.3

Bildet man eine dichotome Variable, ob in der Familie Deutsch oder eine andere Sprache gesprochen wird, so ergibt sich bei der Skala Zukunftsperspektive folgender signifikanter Effekt (Mittlerer Rang Deutsch als Familiensprache 331, andere Sprache 394, $p < .05$, $Z = -1.99$). Bei den anderen drei Skalen sind keine signifikanten Unterschiede festzustellen.

Vergleicht man die Gruppen der Teilnehmenden, die einen eigenen Rechner besitzen mit denen, die keinen eigenen Computer besitzen, so ergibt sich ein signifikanter Unterschied in allen Skalen: Selbstkonzept ($p < .05$, $Z = -1.93$), Technikexpertise ($p < .01$, $Z = -4.79$), Selbsteinschätzung ($p < .05$, $Z = -1.90$) und Zukunftsperspektive ($p < .1$, $Z = -1.84$). Diejenigen die Angaben einen eigenen Rechner zu Hause zu haben, verfügen auf allen Skalen also über signifikant positiverer Ausprägungen.

Wirkung des Workshops

Zur Beantwortung der Frage nach der Wirksamkeit eines Workshops betrachten wir die Veränderungen direkt vor und unmittelbar nach dem Kurs (Tabelle 2.4). Dabei bezeichnet „Negative Ränge“ eine Abnahme in den Skalen, Positive Ränge demnach eine Zunahme in den einzelnen Skalen und Bindungen keine Veränderung.

Ränge insgesamt

		N	Mittlerer Rang	Rangsumme
Einschätzung nach –	Negative Ränge	163	237,40	38695,50
Einschätzung vor	Positive Ränge	343	261,15	89575,50
	Bindungen	162		

Selbstkonzept Nach – Selbstkonzept vor	Negative Ränge	117	211,57	24753,50
	Positive Ränge	408	277,75	113321,50
	Bindungen	144		
Zukunftsswunsch Nach – Zukunftsswunsch Vor	Negative Ränge	142	212,86	30226,50
	Positive Ränge	313	234,87	73513,50
	Bindungen	176		

Tabelle 2.4

Die Veränderung der Selbsteinschätzung, des Selbstkonzepts und der Zukunftsperspektive ist insgesamt signifikant positiv. Nach Wilcoxons Rangsummentest:

- Selbstkonzept Positive Ränge: 408 von 669, $p < .01$, $Z = -12,9$,
- Zukunftsperspektive Positive Ränge: 313 von 631, $p < .01$, $Z = -7,8$,
- Einschätzung Positive Ränge: 343 von 668, $p < .01$, $Z = -7,8$.

Weder die unterschiedlichen Altersgruppen noch die Differenzierung nach Geschlecht sind in den obigen Skalen unterscheidbar. Auf die Gesamtgruppe wirkt die Maßnahme signifikant positiv auf die Skalen Selbstkonzept, Selbsteinschätzung und Zukunftsperspektive. Jedoch haben die Workshops nicht auf alle Kinder eine positive Wirkung unabhängig von Geschlecht und Alter.

Zur Untersuchung, ob erfolgreiche außerschulische monoedukative Maßnahmen auch in einem koedukativen Klassenverband gleiche Ergebnisse erzielen können, haben wir im zweiten Schritt eine kontrollierte Studie mit Schulklassen der Region Aachen durchgeführt.

3. Kontrollierte Studie

Die Ergebnisse basieren im Gegensatz zu den oben beschriebenen Beobachtungen nicht auf der Befragung der überwiegend freiwillig teilnehmenden Schülerinnen und Schüler, sondern auf einer Zufallsstichprobe. Hierdurch können verschiedene Einflüsse wie Alterseffekte oder ein Effekt durch die freiwillige Teilnahme am Kurs minimiert werden.

Die Studie wurde als Teil einer Längsschnittstudie¹⁰ über einen Zeitraum von drei Monaten mit drei Versuchsgruppen durchgeführt. Die Versuchsgruppen setzen sich aus einer Experimentalgruppe (Klasse A, B und C) sowie einer Kontrollgruppe (Klasse D) zusammen. Es handelt sich um Schüler und Schülerinnen der sechsten Klassenstufe aus drei Gymnasien; aus schulorganisatorischen Gründen war es nicht möglich randomisierte Versuchsgruppen zu bilden.

An der Erhebung haben insgesamt 81 Schülerinnen und Schüler teilgenommen. Davon waren 35 weiblich (43%) und 46 männlich (57%). Alle Schülerinnen und Schüler besuchten die sechste Klasse. 33 Schülerinnen und Schüler waren elf Jahre alt, 47 zwölf Jahre alt und ein Schüler war 13. Im arithmetischen Mittel waren die Schülerinnen und Schüler ca. 11,6 Jahre alt.

		Alter			Gesamt
		11	12	13	
Geschlecht	Weiblich	12	23	0	35
	Männlich	21	24	1	46
Gesamt		33	47	1	81

Tabelle 3.1

Einstellungen der Versuchsgruppe

Nur ein Teilnehmer hat zuhause keinen Zugang zu einem Computer. Fast 70% gab an, über einen *eigenen* Computer zu verfügen. Im Gegensatz zu der Stichprobe aus den *go4IT!*-Roboter-Workshops lässt sich in der Stichprobe der Experimentalgruppe kein Einfluss eines eigenen Computers auf den Zukunftsperspektive ($Z = -1,549, n.s.$), das Selbstkonzept ($Z = -1,154, n.s.$), die Selbsteinschätzung ($Z = -,413, n.s.$) oder die Technikexpertise ($Z = -1,255, n.s.$) ableiten.

86% der Teilnehmenden gaben Deutsch als Familiensprache an. Weitere Familiensprachen sind Russisch (2,7%), Englisch und Vietnamesisch (1,8%), sowie Polnisch, Italienisch, Chinesisch und Kantonesisch (0,9%). Es zeigt sich kein Einfluss der Familiensprache auf den Zukunftsperspektive der Schülerinnen und Schüler ($Z = -,407, n.s.$), deren Selbstkonzept ($Z = -1.157, n.s.$), die Selbsteinschätzung ($Z = -1,169, n.s.$) und die Technikexpertise ($Z = -0.107, n.s.$). Daher wird im Folgenden die Familiensprache als Erfolgsfaktor nicht weiter betrachtet.

¹⁰ Mit der Experimentalgruppe wurde ein zweitägiger Roboterworkshop (singuläre Maßnahme) durchgeführt. Klasse D fungierte als Kontrollgruppe. Die Auswertung der Längsschnittstudie befindet sich noch in der Auswertung und ist nicht Teil dieses Artikels.

Das Geschlecht der Teilnehmenden beeinflusst signifikant die Selbsteinschätzung ($Z = -2,473$, $p < .05^*$), die Technikexpertise ($Z = -2,993$, $p < .05^*$) und den Wunsch, zukünftig eine Karriere im MINT-Bereich anzustreben ($Z = -3,867$, $p < .05^*$). Beim Selbstkonzept ist eine Beeinflussung durch das Geschlecht zu vermuten, auch wenn das Signifikanzniveau von $p = .05$ knapp verfehlt wurde ($Z = -1,911$, $p = .056 > .05^{(*)}$).

Es zeigt sich ein deutlicher Effekt des Geschlechts auf die Kontrollüberzeugung im Umgang mit Technik ($Z = -1,306$, $p < .01^{**}$). Die Kontrollüberzeugung im Umgang mit Technik ist bei den Mädchen mit einem Mittleren Rang von 29.1 deutlich geringer ausgeprägt als die der Jungen mit einem mittleren Rang von 48.2. Unsere Erhebung steht damit im Einklang zu vergleichbaren Studien, die diese Variable betrachtet [ZJ09, BL10] haben.

Wirkung des Workshops

Das Selbstkonzept der Schülerinnen und Schüler ist in der Abschlusserhebung signifikant höher als in der Erhebung zu Beginn des Workshops ($Z = -4,005$, $p < .01^{**}$). Lediglich 12 Schülerinnen und Schüler haben sich im Verlauf des Workshops verschlechtert, bei 23 gab es keine Veränderung im Selbstkonzept und bei 42 Schülerinnen ließ sich im Anschluss ein höheres Selbstkonzept feststellen.

Ebenso steigt die Selbsteinschätzung der Teilnehmenden ($Z = -3,416$, $p < .01^{**}$). 16 Teilnehmende verringerten ihre Selbsteinschätzung, bei 13 veränderte sich die Selbsteinschätzung nicht und bei 49 Schülerinnen und Schülern stieg die Selbsteinschätzung zum Ende des Kurses. Analog verändern die Schülerinnen und Schüler ihre Zukunftsperspektive zugunsten des MINT-Bereichs ($Z = -1,990$, $p < .05^*$).

Es zeigt sich ebenso wie bei der *go4IT!*-Stichprobe kein Effekt des Geschlechts auf die Wirkung des Kurses. So haben die Mädchen am Anschluss des Kurses eine signifikant höhere Selbsteinschätzung ($Z = -1,979$, $p < .05^*$) und ein höheres Selbstkonzept ($Z = -2,760$, $p < .05^*$). 18 Schülerinnen konnten durch den Kurs verbessern, bei 9 ließ sich keine Veränderung feststellen und lediglich 9 haben sich im Selbstkonzept verschlechtert. Bei der Selbsteinschätzung zeigte sich bei 20 Schülerinnen eine Verbesserung, bei 8 eine Verschlechterung und bei 5 Schülerinnen fand keine Veränderung statt.

Ebenso wirkt der Kurs signifikant positiv auf das Selbstkonzept ($Z = -2,910$, $p < .05^*$) und die Selbsteinschätzung ($Z = -2,843$, $p < .05^*$) der Jungen. Lediglich bei 6 Schülern wurde eine Verschlechterung des Selbstkonzepts festgestellt, wohingegen 24 Jungen ihr Selbstkonzept steigerten. Bei 14 Jungen fand keine Änderung des Selbstkonzepts statt. Analog nahm die Selbsteinschätzung bei nur 8 Jungen ab und steigerte sich bei 24 Jungen. Die Veränderung der Zukunftsperspektive lässt sich für die Geschlechter nicht belegen. Weder die Jungen ($Z = -1,412$, *n.s.*), noch die Mädchen ($Z = -1,421$, *n.s.*) verändern ihre Zukunftsperspektive signifikant.

4 Zusammenfassung und Ausblick

Es zeigt sich, dass die Versuchspopulation der kontrollierten Studie in ihren Einstellungen im Wesentlichen mit den im Projekt erhobenen übereinstimmt. Bei der Versuchsgruppe der Teilnehmenden aus den go4IT! – Kursen zeigen die 12-jährigen eine signifikant positivere Selbsteinschätzung gegenüber Informatik und Technik Inhalten als die 11-Jährigen. Da in NRW aber keine schulische Förderung in diesem Bereich in der Altersstufe stattfindet, ist eine allgemeine Steigerung des Selbstkonzeptes in diesem Alter anzunehmen. Auf den anderen drei Skalen ist kein Alterseffekt messbar. Betrachtet man die Jungen und Mädchen getrennt, so geben die Jungen ein signifikant positiveres Selbstkonzept, eine höhere Technikexpertise, eine positivere Selbsteinschätzung an und können sich eher eine Zukunft im Informatik und Technikbereich vorstellen. Diese Ergebnisse zeigen, dass ein zu vermutendes Auseinandergehen der Geschlechter bei der Einstellung gegenüber Informatik und Technik schon vor dem 11ten Lebensjahr geschieht. Der Zugang zu einem eigenen Rechner zu Hause hat bei der go4IT! – Versuchsgruppe einen signifikant positiven Einfluss auf alle erhobenen Skalen. Im Gegensatz dazu lässt sich in der Stichprobe der Experimentalgruppe kein signifikanter Einfluss feststellen. Da weder bei den Mädchen noch den Jungen in der go4IT!-Versuchsgruppe als einzelne Gruppe betrachtet signifikante Unterschiede auftreten, wird der Unterschied in der Stichprobengröße der Experimentalgruppe vermutet. Die Unterscheidung Deutsch als Familiensprache oder eine andere Sprache ergab nur einen signifikanten Unterschied. So halten die Teilnehmenden mit nicht deutsch-sprachigem Hintergrund - entgegen der bildungspolitisch geprägten Erwartung - eine spätere Zukunft im Informatik und Technik Bereich eher für möglich als die deutschsprachigen Teilnehmenden. Dies unterstützt die Hypothese, dass die negative Einstellung gegenüber Informatik und Technik ein kulturelles Problem speziell in den westlichen Ländern ist.

Betrachtet man die Wirkung des Kurses, so wirkt die Maßnahme in der go4IT! Versuchsgruppe sowie in der Experimentalgruppe signifikant positiv auf die Skalen Selbstkonzept und Selbsteinschätzung. Die Veränderung der Zukunftsperspektive lässt sich bei der Experimentalgruppe nicht belegen. Die Wirkung des Kurses auf Selbstkonzept und Selbsteinschätzung lässt sich damit auf koedukative Kurse und Klassenverbände erfolgreich übertragen. Wenn schulorganisatorisch möglich, können erfolgreiche Technikinteresse-initiiierende Maßnahmen ein wichtiger Bestandteil der allgemeinen schulischen Ausbildung werden.

Als nächste Schritte werden wir die Nachhaltigkeit der Veränderung der Teilnehmenden auf den positiv beeinflussten Skalen Selbstkonzept, Zukunftsperspektive und Selbsteinschätzung messen, indem Nachfolge Workshop in unterschiedlichen Zeiträumen angeboten werden und wiederum eine Maßnahme durchgeführt und evaluiert wird.

Literaturverzeichnis

- [AZ07] Arning, K.; Ziefle, M.: Understanding age differences in PDA acceptance and performance. *Computers in Human Behavior*, 23 (6), 2904–2927, 2007.
- [BE99] Beier G. :Kontrollüberzeugungen im Umgang mit Technik. *Report Psychologie*. 24(9), S. 684-693, 1999.
- [BL10] Brauner, P; Leonhardt, T; Ziefle, M; Schroeder, U.: The effect of tangible artifacts, gender and subjective technical competence on teaching programming to seventh graders. In *Proceedings of the 4th International Conference on Informatics in Secondary Schools: Evolution and Perspective 2010 (ISSEP)*, Zurich, 2010.
- [BR98] Brosnan, M.J.: The impact of computer anxiety and self-efficacy upon performance. *Journal of Computer Assisted Learning*, 14, S. 223–234., 1998.
- [BU95] Busch T. Gender differences in self-efficacy and attitudes toward computers. *Journal of Educational Computing Research*.12, S. 147-158, 1995.
- [FH96] Funken, C.; Hammerich, K.; Schinzel, B.: *Geschlecht, Informatik und Schule. Oder: Wie Ungleichheit der Geschlechter durch Koedukation neu organisiert wird.* Sankt Augustin: Academia Verlag 1996.
- [HS05] Hartmann, S.; Schecker, H.: Bietet Robotik Mädchen einen Zugang zur Informatik, Technik und Naturwissenschaft? – Evaluationsergebnisse zu dem Projekt „Roberta“. In: *Zeitschrift für Didaktik der Naturwissenschaften*, Jg. 11, 2005.
- [LG03] Liu, L.; Grandon, E.E: How performance and self-efficacy influence the ease of use of object-orientation. *Proceedings of the 36th Hawaii International Conference on System Science (HICSS'03)*, January 2003, Big Island, HI, S. 327–336, 2003.
- [LS09] Leonhardt, T.; Schroeder, U.: go4IT!: Initiierung und nachhaltige Förderung von Interesse an MINT-Fächern bei Mädchen. In *Informatische Bildung in Theorie und Praxis, Beiträge zur INFOS 2009*, 13. GI-Fachtagung - Informatik und Schule, Berlin, S. 132-142, 2009.
- [PT07] Petersen, U.; Theidig, G. et al.: Roberta Abschlussbericht, Gefördert vom Bundesministerium für Bildung und Forschung, Laufzeit 1.11.2002 – 28.2.2007, http://www.iais.fraunhofer.de/fileadmin/images/pics/Abteilungen/AR/PDF/Abschlussbericht_Roberta_2007-11-21.pdf, 2007.
- [SS05] Schelhowe, H.; Schecker, H.: Wissenschaftliche Begleitung des Projekts ROBERTA — Mädchen erobern Roboter. Abschlussbericht Berichtszeitraum 1.11.2002 - 30.10.2005. In: [PET07], 2005.
- [ZJ09] Ziefle M, Jakobs E-M. *Wege zur Technikfaszination - Sozialisationsverläufe und Interventionszeitpunkte*. Berlin: Acatech, Springer; 2009.
- [WE99] Westram, H.: *Schule und das neue Medium Internet; wie gehen die beiden Geschlechter damit um?* Dissertation, Universität Dortmund 1999.

Informatik begreifen – Zur Nutzung von Veranschaulichungen im Informatikunterricht

Manuela Kalbitz, Hendrik Voss, Carsten Schulte
Königin-Luise Str. 24-26,
Freie Universität Berlin
14195 Berlin

Abstract: Viele informatische Lerngegenstände sind abstrakt, d.h. nicht direkt beobachtbar, und werden daher im Unterricht formal bzw. symbolhaft dargestellt. Das gilt für Algorithmen, Funktionsprinzipien von Hardware usw. Daher sind Veranschaulichungshilfen relevant. Ausgehend von Bruner lassen sich auf dem Weg zur Symbolebene enaktive und ikonische Veranschaulichungen als Hilfsmittel zum Be-Greifen unterscheiden. Im Beitrag gehen wir in einer nicht-repräsentativen und eher qualitativ angelegten empirischen Studie der Frage nach, wie weit diese drei Repräsentationsebenen im Informatikunterricht verwendet werden. Dazu gehen wir zunächst auf die lerntheoretischen Hintergründe ein. Anschließend präsentieren wir dann die Umfrage und deren Ergebnisse.

1 Einführung

Informatik, als Wissenschaft der „automatischen Verarbeitung von Information“, befasst sich mit symbolhaft repräsentierter Information. Der Zugang zu den dabei verwendeten abstrakten, arbiträren und formalen Notationen und das Verstehen der zugeordneten Semantik sind nicht immer einfach. Daher werden etwa in der Modellierung auch halb- oder in-formale Notationen verwendet; zum Teil werden Informationen sogar durch Handlungen wie Rollenspiele ausgedrückt. Informatische Konzepte zu erlernen, bedeutet also zumeist Zugang zu und Verstehen von symbolhaft repräsentierter Information. Um diesen Zugang zu erleichtern, kann die symbolische Darstellungsebene durch in- und halbformale (=ikonische) oder auch handelnde (=enaktive) Darstellungen ergänzt werden. Zurückgehend auf den Psychologen Bruner ist dieses Verfahren auch als EIS-Prinzip bekannt (EIS für den Wechsel von *Enaktiv*, *Ikonisch* und *Symbolisch*).

In diesem Artikel gehen wir in einer nicht-repräsentativen und eher qualitativ angelegten empirischen Studie der Frage nach, wie weit dieses EIS-Prinzip im Informatikunterricht verwendet wird. Dazu gehen wir zunächst auf die lerntheoretischen Hintergründe ein. Anschließend erläutern wir dann den Aufbau der Studie und präsentieren deren Ergebnisse.

2 (Lern-)theoretische Fundierung

Wie sich die Art des Denkens eines Kindes im Laufe der Jahre verändert, hat der Schweizer Entwicklungspsychologe Jean Piaget (1896 - 1980) erforscht. Für die Didak-

tik interessant sind vor allem drei Hauptstadien, die sich wesentlich voneinander unterscheiden (der folgende Abschnitt nach [Ze98], S. 89 ff.):

Zunächst ist das Denken von Zwei- bis Sechsjährigen im *präoperationalen Stadium* durch konkrete Handlungen an Objekten geprägt. Zum Beispiel können sie nur handhabbare Objekte – etwa Puppen – in ihrer Größe vergleichen, während sie keine gezeichneten Streifen in eine gedankliche Reihenfolge von klein nach groß sortieren können. Die Erfahrungen der unmittelbaren Anschauung können erst Kinder im Grundschulalter gedanklich zusammenfügen oder umkehren. Piaget spricht von dem *Stadium der konkreten Operationen*. Der Größenvergleich von Puppen kann nun aus der eigenen Imagination heraus auf gezeichnete Streifen übertragen werden – einzelne Vergleiche werden in Gedanken zu einer Reihung zusammengesetzt. Erst ab etwa zwölf Jahren vermögen es Kinder, auch aus Sprache und Symbolen abstrakte Sachverhalte zu erschließen. In diesem *Stadium der formalen Operationen* sind etwa Größenvergleiche so weit verinnerlicht, dass auch Aufgaben der Art „a ist kleiner als b und b ist kleiner als c, was ist am kleinsten?“ gelöst werden können. Diese Erkenntnisse verarbeitete der US-amerikanische Psychologe Jérôme Bruner (1915) zu seiner Theorie der drei Darstellungsebenen, welche den drei Stadien Piagets entsprechen. So lernt der Mensch *enaktiv* durch das Experimentieren am konkreten Material – er begreift also im wörtlichen Sinne seine Umwelt. Zweitens erkennt er Sachverhalte durch Bilder und Zeichnungen, welche als *ikonische* Darstellungen bezeichnet werden. Und schließlich gibt es noch die *symbolische* Ebene, in welcher Erkenntnisse gewonnen werden können. Sie umfasst nicht nur Zeichensysteme, sondern auch verbale Mitteilungen. Schließlich zeichnet sich nach Bruner ein intellektueller Erwachsener dadurch aus, dass er flexibel zwischen den Darstellungsebenen wechseln kann.

In den 1970er-Jahren war der Umgang mit Computern auf die symbolische Ebene beschränkt. Informatikern wie Alan Kay war klar, dass die Arbeitsweise von Computern für Laien zu kompliziert ist, um sie lediglich symbolisch darzubieten. So hat er bereits bei der Entwicklung des Desktops versucht, die symbolische Ebene durch eine ikonische (bildliche) Oberfläche mit „Fenstern“ zu ergänzen, um den Umgang mit einem Computer einfacher zu gestalten ([Ka09], S. 197). Dazu hat er sich bewusst an das EIS-Prinzip angelehnt und darauf aufbauend eine auf den Computer bezogene Darstellungsebene definiert: die enaktiv-ikonische, die das Arbeiten mit der Maus (Zeigen, Auswählen Drag'n'Drop) beschreibt. Eine Art ‚Zwischenebene‘, denn obwohl das Umgehen mit der Maus motorische Handlung ist, kann sie nicht unbedingt als enaktiver Zugang zur Wirklichkeit bzw. zum Gegenstand verstanden werden. Aktuelle Touchscreens verschieben diese Ebene durch haptisches Feedback und den Ersatz der Maus durch den Finger weiter in Richtung der enaktiven Ebene. Doch gleich, ob nun drei oder vier Ebenen unterschieden werden, didaktisch bedeutsam ist der Wechsel. Darauf weisen verschiedene Arbeiten hin: Wird ein Unterrichtsstoff von Beginn an ausschließlich durch Symbolik und Sprache gelehrt, fällt die Verknüpfung mit bereits Bekanntem schwer. Denn Symbolsprachen sind häufig unbekannt und nicht mit Erfahrungen verbunden.

Gerade dies ist aber, wie die empirische Forschung bzw. die Gehirnforschung bestätigt, für langfristige Verbindungen von Neuronen – also einem nachhaltigen Verinnerlichen – von großer Bedeutung ([Kl08], S. 15). Correll sieht gar einen Zusammenhang zwischen der unangemessenen Darbietung von Lernstoff und Lernstörungen ([Co89], S. 50). Und

die „Erleuchtung“ oder „Einsicht“, die Erwachsene oft haben, sei nichts anderes als das Ergebnis von abtastenden Versuchen auf der Vorstellungsebene ([Ro69], S. 260). Also können gerade ikonische Visualisierungen dem Vereinfachen und Strukturieren von Inhalten und Prozessen dienen.

Weitere Erkenntnisse der Hirnforschung besagen, dass jeder Mensch auf andere Weise am besten Sinneseindrücke wahrnehmen und verarbeiten kann: entweder durch das Hören, das Sehen oder das eigene Handeln. Guski kommt hingegen zu dem Ergebnis, dass visuell mehr Informationen aufgenommen werden können als auditiv ([Gu89], S. 167 f.). Auch für das Gedächtnis sind Reize auf allen drei Ebenen sinnvoll. So belegen Studien von Schnotz und Bannert, dass auditiv-sprachliche und visuell-bildliche Informationen unterschiedlich abgespeichert und verarbeitet werden. Allerdings kann die gleichzeitige Verarbeitung verschieden dargebotener Informationen auch zu Belastungen des Arbeitsgedächtnisses führen, wenn die Darbietungen nicht direkt das Verständnis fördern [Sc99]. Dennoch behalten Menschen auch laut Gemmer etwa 20 Prozent von dem im Gedächtnis, was sie hören, 30 Prozent von dem, was sie sehen, und 90 Prozent von dem, was sie unter Einsatz unterschiedlicher Sinne selbst tun ([Ge04], S. 74).

Nicht nur aus empirischer Sicht, auch aus bildungstheoretischer Sicht wird das EIS-Prinzip als bedeutsam eingeschätzt: W. Klafki [Kl07] diskutiert die Relevanz verschiedener Repräsentationsweisen im Zusammenhang mit dem Prinzip des exemplarischen Lernens und stellt dabei vier seiner Meinung nach wesentliche Gesichtspunkte heraus: 1) Die Repräsentationsebenen bauen aufeinander auf. Die höhere ist auf die vorausgehende zwingend angewiesen. 2) Das bedeutet jedoch nicht, dass die drei Stufen in jedem „besonderen Lernakt“ vollzogen werden müssen ([Kl07], S. 158). 3) Oft sind zwei der drei Stufen eng miteinander verzahnt und 4) ab dem 10-12 Lebensjahr kann man in wachsendem Maße „symbolische geistige Akte erwarten“ ([Kl07], S. 158). Aber auch für Erwachsene gelte, dass die ersten beiden Stufen eine große Bedeutung besitzen. Klafki schlussfolgert: „Einer der gravierenden Mängel unseres üblichen Schulunterrichts in allen Schulformen und auf allen Schulstufen dürfte darin liegen, daß eben dieser Sachverhalt vielfach verkannt wird und daß verstehendes/entdeckendes Lernen gerade auch auf der abstrakt-symbolischen Stufe geradezu verhindert wird, weil man zu früh und zu ausschließlich auf dieser Ebene ansetzt“ ([Kl07], S. 159).

Nicht zuletzt wird das EIS-Prinzip auch in der Informatikdidaktik diskutiert. Auf die Bedeutung des EIS-Prinzips für den Informatikunterricht weisen verschiedene Autoren hin: Humbert betont, dass „im Zusammenhang mit dem EIS-Prinzip dem >>E<< im Informatikunterricht eine besondere Rolle zukommt [...]“, denn „gerade abstrakte Gegenstände [sollten] enaktiv zugänglich gemacht werden“ ([Hu06], S. 78). Hartmann et.al. behaupten: „Die enaktive Repräsentationsform eignet sich besonders für den Einstieg in ein Thema. Der Stoff wird für die Lernenden zugänglicher und besser im Gedächtnis verankert.“ ([HNR07], S.116) Allerdings sei „der Vorbereitungsaufwand für enaktive Repräsentationen [...] nicht zu unterschätzen und muss optimiert werden.“ ([HNR07], S.117) Zuletzt betonte Jürgen Müller im LOG IN Heft Nr. 160/161, dass symbolische Modelldarstellungen auch im Informatikunterricht oft erhebliche Schwierigkeiten bereiten und daher gegenständliche Realisationen und ikonische Darstellungen „von besonderer Wichtigkeit“ sind ([Mü09], S. 24). Schließlich führt er gemeinsam mit Andreas Schwill Ideen für gegenständliche Methoden an.

Aus der Literaturanalyse schlussfolgern wir, dass die Repräsentationsebenen auch für den Informatikunterricht bedeutsam sind. Ein Arbeiten nach dem EIS-Prinzip begünstigt Methodenvielfalt und Motivation. Die verschiedenen Methoden - im Wechsel eingesetzt - können Eintönigkeit verhindern und konkretes Material kann Neugier wecken. Der Einsatz eigener motorischer Fähigkeiten spricht speziell handwerklich Begabte an. Darüber hinaus begünstigen speziell die enaktiven Methoden die Aneignung outputbezogener Kompetenzen. Fähigkeiten werden beobachtbar: es ist zu sehen, ob die Schülerinnen und Schüler den Sachgegenstand beherrschen. Und Fehler, die auf der symbolischen Ebene nicht identifiziert werden können, drücken sich mitunter auf der enaktiven ganz anders aus. Man denke hierbei an Verständnisproblemen von objektorientierter Sprache, welche im Rollenspiel, bei dem keine Syntaxprobleme für Verwirrung sorgen können, nachvollziehbar während des Durchlaufs offenbart werden.

Neben Rollenspielen (z.B. [DGZ05], [Fo07]) bietet der Informatikunterricht einige Möglichkeiten für enaktives Arbeiten. Sortialgorithmen können zum Beispiel an konkretem Material, etwa einer Bücherreihe oder einer CD-Sammlung, durchgeführt werden. Roboter können programmiert werden, sodass sie sich entsprechend eines selbst geschriebenen Programms bewegen. Die Auswirkungen des Quelltextes werden direkt sichtbar und können in verschiedenen Raumsituationen getestet werden. Weiterhin ermöglicht das Basteln zum Beispiel eines Wikis aus Papier und Fäden (siehe z.B. [Ho07]) ein Gestalten nach eigenen Vorstellungen, wodurch anfangs gestellte Vermutungen und Erwartungen überprüft und handhabbar gemacht werden.

Uns ist jedoch keine empirische Studie bekannt, die die Verwendung dieses Prinzips im Informatikunterricht untersucht hat.

3 Umfrage zur Nutzung des EIS-Prinzips im Informatikunterricht

Um die Verwendung des EIS-Prinzips zu überprüfen, wird eine Umfrage unter Informatiklehrkräften durchgeführt. Wie im Abschnitt 2 anhand der Literaturdiskussion deutlich wurde, ist das EIS-Prinzip einerseits als relevant für den Informatikunterricht einzuschätzen, andererseits wird es möglicherweise jedoch kaum oder zu wenig angewendet – was eventuell auch an der schwierigen Umsetzung bzw. den fehlenden Materialien liegen könnte – insbesondere für den enaktiven Repräsentationsmodus. Uns interessieren daher die folgenden beiden Fragen: Wie groß ist der Anteil an Lehrerinnen und Lehrern, die regelmäßig das EIS-Prinzip in den meisten Themengebieten anwenden? Wie groß ist das Potential des enaktiven Repräsentationsmodus für eine vermehrte Anwendung im Informatikunterricht?

3.1 Konstruktionsprinzipien für das Instrument

Es stellt sich nun die Frage, wie man die unterrichtliche Verwendung des EIS-Prinzips empirisch erheben kann. Der Schwerpunkt unserer Erhebung sollte auf der relativen Häufigkeit liegen, mit der das EIS-Prinzip im Informatikunterricht angewendet wird; und zwar unabhängig davon, ob der Lehrkraft der Begriff „EIS-Prinzip“ bekannt ist oder

nicht. Daher ist eine Befragung der Lehrer nach den Darstellungsebenen, die sie im Unterricht einsetzen, vonnöten. Falls alle drei Darstellungsebenen innerhalb eines Themengebietes regelmäßig, d.h. über mehrere Jahre hinweg, angewendet werden, kann eine Arbeit nach dem EIS-Prinzip angenommen werden.

Wie kann nach der regelmäßigen Verwendung gefragt werden, ohne große Unsicherheiten oder subjektive Verzerrungen durch die Fragestellung zu bewirken? Beispielsweise könnte durch die Erwähnung des dahinterliegenden Prinzips eine soziale Erwünschtheit in der Beantwortung der Fragen auftauchen, die das Antwortverhalten beeinflusst – so bezeichnet etwa W. Klafki die Nicht-Beachtung des EIS-Prinzips als einen „gravierenden Mangel unseres üblichen Schulunterrichts“ ([Kl07], S. 159).

Ein weiteres Problem: Wie können Deutungsschwierigkeiten der Begriffe „enaktiv“, „ikonisch“, „symbolisch“ oder „Darstellungsebene“ bzw. „Repräsentationsmodus“ vermieden werden? Klafki findet eine prägnante Formulierung für das EIS-Prinzip, welche den Weg für die Konstruktion des Umfrage-Instruments weist. Er beschreibt das EIS-Prinzip als „drei Weisen oder Niveaus, denen entsprechend die Auseinandersetzung mit der Wirklichkeit und ihre Aneignung im Lernprozess erfolgen kann: * In der Weise direkten, handelnden Umgangs mit der Wirklichkeit (i.w.S.d.W.), z.B. im Explorieren und Erproben [...] * Im Medium von Bildern, Schemata, Skizzen, anschaulichen Erzählungen und Berichten, Darstellungen (z.B. im Rollenspiel [...]) * Im Medium abstrakter Begriffe [...], ‘nur noch’ gedanklich vollzogenen Operationen und theoretischen Argumentationen“. (S. 157). Die von Klafki vorgenommene Übertragung der Repräsentationsebenen auf deren Rolle im Lernprozess verdeutlicht, dass diese im Unterricht in Form von verwendeten Medien und Methoden beobachtbar werden. Wir können also ‚neutraler‘ nach verwendeten Methoden und Medien/Materialien anstatt nach einem lerntheoretischen Prinzips fragen. Dies sollte einer Lehrperson ohne allzu große subjektive Verzerrungen möglich sein. Es ergibt sich also eine Matrix mit den Dimensionen „Themengebiet“ und „Methode/Material“, sowie einer Angabe der entsprechenden Häufigkeit. Letztere wurde in vier Stufen eingeteilt: „Nie“, „Selten“, „Oft“ und „Jedes Mal“. Das Eintragen einer der beiden letzten Stufen wird als notwendige Bedingung für eine Verwendung des EIS-Prinzips angesehen.

Um den Fragebogen gut auswertbar zu gestalten und dennoch kurz zu halten, werden verschiedene Einschränkungen vorgenommen. Zunächst werden Themengebiete vorgegeben, in denen das EIS-Prinzip verwendet werden könnte. Dazu wird nach denjenigen gefragt, welche im Berliner Rahmenlehrplan vorgeschrieben sind. Im Lehrplan sind unter anderem die Themengebiete Algorithmen, Automaten, Datenbanken, Kryptologie, Objektorientierung und Sprachen aufgeführt. Da wir voraussetzen möchten, dass symbolische Methoden auf jeden Fall verwendet werden, um sie nicht überprüfen zu müssen, betrachten wir nur Themengebiete, in welchen die symbolische Ebene mit großer Sicherheit ausführlich im Unterricht behandelt wird. Daher werden die Themen „Informatik, Mensch und Gesellschaft“, „Datenschutz“ und „Datensicherheit“ im Fragebogen nicht aufgeführt. Das soll nicht heißen, dass diese Themen belanglos wären oder für diese Themen das EIS-Prinzip nicht gilt. Allerdings erübrigt sich durch diese Beschränkung ein Erfragen von Methoden, die die symbolische Darstellungsebene ansprechen und der Umfang der Matrix wird den Befragten zumutbar.

Für den Fragebogen müssen nun die Darstellungsebenen den Methoden und Medien

zugeordnet werden. Das ist nicht unproblematisch, wie man an der erwähnten Methode des Rollenspiels kurz erläutern kann. Klafki hat es der ikonischen Ebene zugeordnet (siehe Zitat oben), als lebendige Darstellung eines Sachverhalts. Wir haben uns im Gegensatz dazu entschieden, dass Rollenspiel der enaktiven Ebene zuzuordnen, also als direkten und handelnden Umgang mit dem Lerngegenstand. Wir sehen eher die ‚spielenden‘ SuS, während Klafki vermutlich eher an die beobachtenden SuS gedacht hat. Obwohl wir versucht haben, die Ebenen treffend zuzuordnen, haben wir im Zweifelsfall eher die grundlegendere Ebene gewählt. Das Erhebungsinstrument könnte also die enaktive und ikonische Ebene leicht überbewerten. Da wir – ähnlich wie die zitierten Autoren – insgesamt jedoch davon ausgehen, dass diese beiden unteren Ebenen im Informatikunterricht eher zu wenig vorkommen, sollte diese Subjektivität nicht zu einer falschen Bestätigung der Ausgangsvermutung beitragen. Insgesamt werden die enaktive und die ikonische Ebene folgendermaßen für den Informatikunterricht präzisiert: Das Besondere an der enaktiven Ebene ist die Möglichkeit der Interaktion mit den Materialien, sodass dem eigenen Handeln eine direkte Rückmeldung gegeben wird. Daher gehören Applets, welche aufgrund der optischen Aufbereitung zwar als bildlich, sprich ikonisch, eingestuft werden, aber interaktive Abläufe visualisieren können, genau genommen zu einer Zwischenstufe, der enaktiv-ikonischen Ebene. Da dies allerdings eine subjektive Interpretation ist und da wir enaktiv als Arbeit fern vom Computer verstehen, zählen wir Applets im Folgenden zu der ikonischen Ebene. Rein ikonisch sind hingegen Tätigkeiten wie das Zeichnen oder Skizzieren. Die bildhafte Darstellung wird zwar im Prozess erschaffen, erlaubt aber keine Erfahrungen durch reale Handlungen. So sind auch Methoden wie das Auswerten von Zeichnungen und Skizzen, das Anschauen von kurzen Videosequenzen oder längeren Dokumentationsfilmen ikonisch. Deutlich wird dabei auch, dass die Verwendung von Darstellungsebenen mit Unterrichtsmethoden verknüpft ist: Applets werden benutzt, Videos angesehen usw. Im Fragebogen wird daher auch von Methoden gesprochen.

Der resultierende Fragebogen setzt sich aus zwei Tabellen und fünf Fragen zusammen. Die erste Tabelle besteht aus sechs Spalten, in denen die oben genannten Schwerpunktthemen des Informatikunterrichts genannt werden. In den acht Zeilen werden verschiedene Unterrichtsmethoden erwähnt, die in zwei Gruppen (ikonisch und enaktiv) unterteilt werden können. Die Befragten sollen angeben, wie oft sie die jeweiligen Methoden in den verschiedenen Themengebieten verwendet haben (0 oder leere Zelle = nie bis 3 = jedes Mal). Um das Erinnern an entsprechende Methoden zu erleichtern, wird empfohlen, den Bogen spaltenweise auszufüllen – anhand eines Unterrichtsthemas dürften sich Lehrerinnen und Lehrer gut der eingesetzten Methoden entsinnen.

Es folgt eine zweite, kleinere Tabelle, in der den einzelnen Methoden ein „Effektivitätswert“ zugeordnet werden soll (0 = nicht effektiv bis 3 = sehr effektiv). Sollte durch die vorherige Bearbeitung der Tabelle eine Beeinflussung stattfinden, so ist sie nicht negativ zu werten. Die Bewertung einer Methode im Alltag dürfte viel mehr dadurch beeinflusst sein, wie oft sie bereits von der Lehrkraft verwendet wurde, als umgekehrt die Bewertung der Methode die Angabe zur Häufigkeit der Umsetzung beeinflusst. Für ein „alltägliches“ Ergebnis empfiehlt sich also erstere Reihenfolge.

Die abschließenden Fragen, etwa ob das EIS-Prinzip bekannt ist, stehen bewusst am Ende der Befragung. Unseres Erachtens nach könnten Fragen, die vor den Tabellen plat-

ziert werden, ersichtlich machen, worauf der Fragebogen abzielt und somit das Ankreuzverhalten der Lehrerinnen und –lehrer an einem entsprechenden Idealbild von sich selbst anpassen. Auf diese Weise sollte ein möglicher Einfluss durch das Phänomen der „sozialen Erwünschtheit“ im Antwortverhalten minimiert werden.

3.2 Durchführung der Befragung

Die über die Berliner GI-Landesgruppe organisierten Lehrerinnen und Lehrer werden als Stichprobe ausgewählt. Auf der dortigen jährlich stattfindenden Jahrestagung und im E-Mail-Verteiler sind über 300 Personen erreichbar. Diese werden über einen in kurzer Zeit ausfüllbaren Fragebogen, der zumeist vorgegebene Antwortmöglichkeiten bereithält, aber auch einige offene Fragen stellt, befragt. Insgesamt wurden 40 Fragebögen zurückgegeben, die als Grundlage dieser Auswertung dienen.

3.3 Ergebnisse und Auswertung

Die folgende Tabelle zeigt die Ergebnisse im Überblick. In den Zeilen werden die Themenbereiche des Informatikunterrichts aufgeführt, in den Spalten verschiedene Methoden bzw. Medien, die hier im Gegensatz zum Fragebogen in einen enaktiven und ikonischen Bereich aufgeteilt werden. Die Ergebnisse zeigen, dass enaktive Methoden (linke Tabellenhälfte) wesentlich seltener eingesetzt werden als ikonische. Der Durchschnitt aller Werte der enaktiven (3,87) gegenüber dem der ikonischen Methoden (11,17) ist weniger als halb so groß. Vor allem die Themengebiete Datenbanken und Sprachen werden fast gar nicht durch enaktive Methoden unterrichtet.

Darstellungsebene Thema	Enaktiv				Ikonisch			
	Basteln	Nicht elektronische Materialien	Roboter	Rollenspiel	Applets	Zeichnen/Malen	Bilder/Skizzen/Videoclips	Längere Filme
Algorithmen	2	8	11	5	16	10	24	2
Automaten	1	4	8	3	18	13	21	2
Datenbanken	0	5	0	2	2	6	22	3
Kryptologie	7	8	0	2	8	3	15	6
Objektorientierung	1	3	4	9	10	11	21	1
Sprachen	0	3	5	2	7	4	13	2

Tabelle 1: Die Werte geben die Anzahl an, wie viele der Befragten die entsprechende Methode im entsprechenden Themengebiet anwenden. Rot und horizontal straffiert sind Werte unter zehn Prozent aller Befragten, blau sind die Werte über 30 Prozent.

Die Umfrage-Ergebnisse sprechen nicht für einen bewussten Einsatz des EIS-Prinzips. Es scheint so, dass lediglich intuitiv passende Methoden oder Materialien eingesetzt

werden, wie z.B. Verschlüsselungsschablonen in der Kryptologie. Außerdem werden enaktive Methoden von recht wenigen Lehrerinnen und Lehrern häufiger verwendet. Im Gegensatz dazu sind ikonische Methoden weit häufiger im Einsatz, was gerade bei Bildern, Skizzen oder kurzen Videoclips an dem damit verbundenen geringen Zeitaufwand erklärbar ist.

Bei welchen Themen fällt es schwer bzw. leicht, geeignetes Material zu finden? Im Fragebogen haben wir die Befragten gebeten, Themen zu nennen, bei denen es ihnen besonders schwer bzw. besonders leicht fällt, schüleraktivierendes Material zu finden. Abbildung 1 zeigt, dass zu Algorithmen, Objektorientierung und Kryptologie die meisten Lehrerinnen und Lehrer am leichtesten Materialien finden, mit denen gearbeitet werden kann. Wie gesehen, werden in diesen Themengebieten auch am meisten enaktive Methoden eingesetzt. Eine Ausnahme sind die Automaten: Obwohl gerade bei diesem Thema vor allem ikonische Methoden verwendet werden (siehe Tabelle 1), empfinden es 8 der 40 Befragten als schwierig, geeignetes Material zu diesem Thema zu finden.

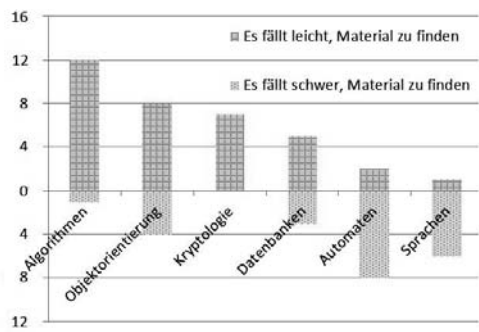


Abbildung 1: Zugänglichkeit geeigneter Unterrichtsmaterialien für die Themengebiete

Umgekehrt verhält es sich mit den Automaten und Sprachen. Die Befragten kennen und verwenden kaum handhabbare Materialien zu den Themen. Insgesamt zeigt sich, dass die meisten Lehrerinnen und Lehrer nur diejenigen enaktiven Methoden kennen, welche sie auch im Unterricht verwenden. Es besteht nach diesem Stand durchaus die Möglichkeit, dass bei einem verbreiteten Wissen zu enaktiven Methoden diese auch vermehrt angewendet werden würden.

Für wie effektiv werden die enaktiven Methoden gehalten? Zunächst sollen hier die Ergebnisse auf die Frage nach der Effektivität der einzelnen Methoden dargestellt werden. Die Ergebnisse sehen im Einzelnen wie folgt aus:

	Skizzen/Bilder/kurze Videoclips (ikonisch)	2,4
	Abläufe visualisierende Applets (ikonisch)	2,1
Nicht elektronische Materialien (enaktiv)		1,7
Roboter (enaktiv)		1,7
Rollenspiel (enaktiv)		1,7
	Zeichnen/Malen (ikonisch)	1,5
	Längere Filme (ikonisch)	1,3
Basteln (enaktiv)		1,0

Tabelle 2: Bewertung der Effektivität (0-3), Mittelwerte

Werden die Methoden, die als effektiv betrachtet werden, wirklich öfter genutzt? Zwischen *kaum effektiv* und *effektiv* wird die Grenze 1,5 für als effektiv betrachtete Methoden gewählt. Daher werden im Folgenden nur die Methoden näher betrachtet, die einen durchschnittlichen Effektivitätswert von 1,5 oder höher haben (Tabelle 3).

Skizzen/Bilder/kurze Videoclips	2,4	Skizzen/Bilder/kurze Videoclips	23
Abläufe visualisierende Applets	2,1	Abläufe visualisierende Applets	17
Nicht elektronische Materialien	1,7	Zeichnen/Malen	11,5
Roboter	1,7	Roboter	9,5
Rollenspiel	1,7	Nicht elektronische Materialien	8
Zeichnen/Malen	1,5	Rollenspiel	7
Tabelle 3: Rangfolge der als effektiv bewerteten Methoden		Tabelle 4: Rangfolge des Methodeneinsatzes	

Nun wird geprüft, ob diese Methoden tatsächlich öfter im Unterricht verwendet werden als die übrigen vorgestellten Methoden. Es werden die Angaben aus Tabelle 1 verwendet, wobei pro Methode lediglich die zwei Spitzenwerte verrechnet werden. Diese Beschränkung wird vorgenommen, da manche Methoden eventuell tatsächlich nicht in allen Themengebieten sinnvoll eingesetzt werden können.¹ (Tabelle 4). Zunächst einmal kann man erkennen, dass die beiden „Spitzenreiter“ des „Effektivitätsrankings“ auch bei der Verwendung im Unterricht ganz vorn stehen (Skizzen/Bilder/Videoclips und Applets). Auch die nicht elektronischen Materialien, die Roboter und das Rollenspiel, stimmen mit der Reihenfolge der oberen Tabelle überein. Die einzige Ausnahme bildet „Zeichnen und Malen“, das als wenig effektiv bewertet wird, aber dennoch häufiger benutzt wird als die drei genannten Methoden. Das bestätigt den oben genannten Gewöhnungseffekt, der sich auf die ikonische Ebene ausrichtet.

4 Fazit

Festzustellen ist, dass bei der Mehrzahl der Informatiklehrerinnen und Informatiklehrern von keinem bewussten Gebrauch des EIS-Prinzips gesprochen werden kann. Obwohl fast ein Drittel der Befragten das Prinzip kennt, ist die Quote derjenigen, die regelmäßig enaktive Methoden und Materialien einsetzen, deutlich geringer. Und auch die ikonischen Methoden werden von höchstens sechzig Prozent häufiger in bestimmten Themengebieten der Informatik angewandt. Insgesamt werden ikonische Methoden innerhalb unserer Angaben eine doppelt so häufige verwendet wie enaktive Methoden. Daher wird gefolgert, dass ein Informatikunterricht nach dem EIS-Prinzip nur selten stattfindet. Zum einen kennen anscheinend viele der Befragten kaum Alternativen. Zum anderen scheint es noch keine Studie zu geben, die die Effektivität des EIS-Prinzips für den Informatikunterricht belegt. Bilder, Skizzen und Videoclips werden am häufigsten regelmäßig eingesetzt. Dabei spielt möglicherweise auch der Zeitaufwand für Vorbereitung, Durchführung und Materialerstellung eine Rolle. Allerdings werden diese Methoden pro

¹ Die Beschränkung erwirkt übrigens keine Änderung der Reihenfolge außer zwischen Robotern und nicht elektronischen Materialien.

Themengebiet nur von etwa der Hälfte aller Befragten genutzt. Unter den enaktiven Methoden werden nur die Standardbeispiele von mehreren Lehrern (um die 20 Prozent) eingesetzt. Übereinstimmung besteht darin, in welchen Themengebieten es leicht- und in welchen es schwerfällt, enaktive Darstellungen zu verwenden. Es scheint also, dass bekannte Darstellungsweisen eingesetzt werden und nicht bewusst das EIS-Prinzip abgelehnt wird.

Insgesamt sind wir optimistisch, dass sich das EIS-Prinzip im Informatikunterricht künftig größerer Beliebtheit erfreuen wird. Dafür ist es jedoch notwendig, das EIS-Prinzip bereits in der Ausbildung der Informatiklehrerinnen und -lehrer fest zu verankern, indem es thematisiert und vor allem erprobt wird.

5 Literaturverzeichnis

- [Co89] Correll, W.: Lernschwächen und Leistungsstörungen erkennen und überwinden. 1989.
- [DGZ05] Diethelm, I.; Geiger, L.; Zündorf, A.: Mit Klebezettel und Augenbinde durch die Objektwelt. In: Friedrich, S. (Hrsg.): Unterrichtskonzepte für informatische Bildung, INFOS 2005. S. 149-160, 2005.
- [Fo07] Fothe, M.: Algorithmen in spielerischer Form. In: Stechert, P. (Hrsg.) Informatische Bildung in der Wissensgesellschaft, S. 31-42, 2007. https://www.unijena.de/unijenamedia/INFOS_Fothe-p-10947.pdf
- [Ge04] Gemmer, B.; Sauer, C.; Konnertz D.: Mind Mapping. Klett, 2004.
- [GI08] GI e.V. (Hg.): Grundsätze und Standards für die Informatik in der Schule. Bildungsstandards Informatik für die Sekundarstufe I. 2008
- [Gu89] Guski, R.: Wahrnehmung: eine Einführung in die Psychologie der menschlichen Informationsaufnahme, Bd. 7. In: Guski, R.; Selg, H. (Hrsg.): Grundriss der Psychologie: eine Reihe in 22 Bänden. Kohlhammer, 1989.
- [Ho07] Honegger, B. D.; Wiki und die fundamentalen Ideen der Informatik. In: Schubert, S.: Didaktik der Informatik in Theorie und Praxis, INFOS 2007. S. 207-216, 2007.
- [HRN07] Hartmann, W.; Näf, M.; Reichert, R.: Informatikunterricht planen und durchführen. 2007
- [Hu06] Humbert, L.: Didaktik der Informatik. 2. Aufl. 2006.
- [Ka90] Kay, A.: User Interface: A Personal View. In: Laurel, B.: The Art of human-computer interface design, 1990.
- [Kl07] Klafki, W.: Neue Studien zur Bildungstheorie und Didaktik. Zeitgemäße Allgemeinbildung und kritisch-konstruktive Didaktik. Beltz, 6. Aufl, 2007
- [Kl08] Klippert, H.: Planspiele: 10 Spielvorlagen zum sozialen, politischen und methodischen lernen in Gruppen. 2008.
- [Mü09] Müller, J.: Einsatz von Modellen in der informatischen Bildung. Beilage zu LOG IN, 29. Jg., Heft Nr. 160/161, 2009.
- [Po07] Poth, O.: Parameter - Untersuchungen zur Vermeidung verbreiteter Fehlvorstellungen im Informatikunterricht der gymnasialen Oberstufe. Schriftliche Hausarbeit. Studienseminar für Lehrämter an Schulen Hamm, 2007. <http://www.ham.nw.schule.de/pub/bscw.cgi/d675586/Examensarbeit-Poth.pdf>
- [Ro69] Roth, H.: Pädagogische Psychologie des Lehrens und Lernens. 11. Aufl., 1969.
- [Sc99] Schnotz, W., Bannert, M.: Einflüsse der Visualisierungsform auf die Konstruktion mentaler Modelle beim Bild- und Textverstehen. Zeitschrift für experimentelle Psychologie 46, S. 216-235, 1999.
- [SLB09] Seyfahrt, T.; Leibfritz, J.; Blunck, A.: Lernen – erinnern – vergessen. 2009.
- [Ze98] Zech, F.: Grundkurs Mathematikdidaktik. 9. Aufl. Beltz, 1998.

Dramatisieren und literarisches Programmieren

Michael Weigend

Institut für Didaktik der Mathematik und der Informatik
Westfälische Wilhelms-Universität Münster
Fliednerstr. 21
48149 Münster
michael.weigend@uni-muenster.de

Abstract: In einer Studie mit 119 Teilnehmern wurde untersucht, inwieweit Schülerinnen und Schüler im Alter von 14 bis 18 ohne spezielle Informatik-Vorkenntnisse metaphorisch formulierte Algorithmen nachvollziehen können und selbst Metaphern zur Verschriftlichung eigener Anleitungen verwenden. Die Kompetenz, eine ansprechende und verständliche literarische Form für einen Algorithmus zu finden, ist eine Kompetenz, die zur Allgemeinbildung gehört und für informatisches Modellieren nützlich ist.

1 Algorithmus und Drama

Ein Computerprogramm kann man sich als abstrakte Form eines Dramas vorstellen. Da werden Akteure beschrieben und Operationen, die sie in bestimmten Situationen ausführen. Freilich verläuft jede Ausführung etwas anders. Es ist wie beim Improvisationstheater. Vorgegeben ist ein allgemeines Muster, etwa eine Rollenaufteilung. Die konkrete Ausführung hängt in der Regel auch von unvorhersehbaren Ereignissen oder Daten ab, die aus der Umgebung stammen. Eine Reihe von Autoren, allen voran Donald Knuth, sehen Computerprogramme als Literatur. „I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature.” [Kn84, S. 99]. Schließlich ist ein Programm ein Text, der zwar auch von Maschinen interpretiert werden kann, der aber im Wesentlichen für Menschen geschrieben ist und von Menschen verstanden werden muss. Knuth prägte für diese Sichtweise den Begriff „literarisches Programmieren“ (literate programming). Claudia Herbst [He02] kritisiert, dass Programme nur von wenigen (meist männlichen) Eingeweihten produziert und verstanden werden. „Among code’s most defining characteristics are its inaccessibility and covert nature.” [He02, S.3]. Zumindest im Open Source-Bereich trifft diese Behauptung nicht zu. Offener Programmtext wird veröffentlicht und muss sich auf einem „Literaturmarkt“ eigener Art behaupten. Beispiel: Im Python Package Index, einem öffentlichen Repository für Python Software, waren am 28. Januar 2011 13092 Python Module gespeichert (<http://pypi.python.org/pypi>). In den Beschreibungen der Module werden sogar Zielgruppen (intended audience) angegeben (Juristen, Wissenschaftler etc.). Wenn Open Source-Programmtexte in der Konkurrenz erfolgreich sein sollen, müssen sie gut formuliert sein. Denn der Leser (Programmierer) hat die Wahl und wird das Produkt nehmen, das er oder sie am besten versteht. Was für den

Quelltext einer Software gilt, gilt erst recht für die Dokumentation. Sie soll im Großen die Struktur und Arbeitsweise einer kompletten Software erklären (Projektmetapher) und im Kleinen (meist in Form von Kommentaren) algorithmische Ideen einzelner Funktionen vermitteln.

1.1 Was ist ein dramatisierter Algorithmus?

Aristoteles [Ar69] beschreibt in seinen Ausführungen zur Tragödie sechs Bestandteile eines Dramas. Dazu gehören die *Handlung*, die ein in sich geschlossenes Ganzes bildet und *Charaktere*, die die Handlung ausführen. Die Charaktere sind so gewählt, dass die Handlung nicht willkürlich erscheint sondern „schicksalhaft“ und als logische Folge festgelegter Charaktermerkmale und Verhaltensdispositionen der Akteure.

In einem dramatisierten Algorithmus sind die Akteure und Handlungen Metaphern für abstrakte algorithmische Elemente. In der Rhetorik versteht man unter einer Metapher die Übertragung eines Inhalts aus einem Wissensbereich (Quelldomäne) auf ein völlig anderes Gebiet (Zieldomäne). Man verwendet Metaphern um einen Text sprachlich interessanter und abwechslungsreicher zu machen [Ba05].

Wenn die Metapher aus einer dem Leser vertrauten Domäne stammt, kann sie auch zum Verständnis beitragen. Der Rezipient kann dann sein bereits vorhandenes Wissen auf eine neue Domäne anwenden. Eine LIFO-Datenstruktur (last in first out) wird metaphorisch als Schlange bezeichnet. Man stellt sich eine Warteschlange an einer Supermarktkasse vor und kann dieses Wissen direkt verwenden um die Funktionalität der Schlange zu verstehen.

Nun kann man auch Aktivität abstrakter geometrischer Gebilde beschreiben: „Die Kreisfläche bewegt sich nach links.“ Wozu also Metaphern? Folgt man der aristotelischen Idee des Dramas, muss die Aktivität in der Beschaffenheit des Akteurs ihren Ursprung haben. Einer Kreisfläche sieht man es nicht an, dass sie ihre Position verändern kann. Bezeichnet man sie metaphorisch als Schildkröte (also als bewegliches Lebewesen), so impliziert dies bereits die Möglichkeit einer Ortsveränderung.

Ein dramatisierter Algorithmus enthält oft auch dekorative Elemente, die das Szenario auskleiden aber kein Pendant im Algorithmus haben. Man könnte sie also weglassen ohne die algorithmische Struktur zu verfälschen. Manchmal sind hinzuerfundene Elemente aber wichtig, um Einzelteile zu einer geschlossenen Gestalt (einer plausiblen Handlung) abzurunden. Abbildung 1 stammt aus einer Aufgabe, die in der Studie verwendet wurde, die später in diesem Beitrag vorgestellt wird.

Das Bild zeigt eine Anordnung aus Formen, die so verändert werden soll, dass Buchstaben entstehen (hier: TEL).

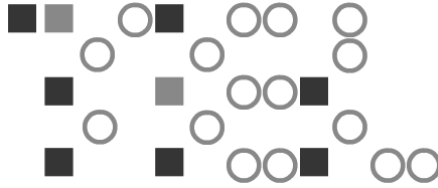


Abbildung 1: Eine Anordnung von Formen, die durch einen Algorithmus verändert wird.

Diese Veränderung wird durch einen einfachen Algorithmus beschrieben:

Am Strand liegen eckige Steine und runde Schildkröten. Jemand ruft „Schildkröten, nach links!“ Und alle Schildkröten gehen ein Stück nach links.

Dies ist ein dramatisierter Algorithmus. Er enthält Metaphern und eine (unscheinbare) Dekoration. Offenbar sind die Steine und Schildkröten Metaphern für Quadrate und Kreise. Der Begriff „Strand“, der Ort des Geschehens, ist für den *Algorithmus* unwichtig. Für das *Drama* ist er jedoch durchaus relevant, weil er die Handlung noch weiter konkretisiert. Man könnte noch weitere Elemente erfinden, etwa eine Ursache für das Geschehen. Eine nicht-dramatisierte Version lautet: *Schiebe alle Kreise ein Stück nach links.*

1.2 Dramatisierung und Software-Entwicklung

In jedem Styleguide findet man den Grundsatz, dass man sinnvolle („sprechende“) Bezeichner wählen soll. Die Bezeichnerwahl geht häufig mit einer Dramatisierung einher, die quasi neben der rein technischen Struktur- und Algorithmus-orientierten Programmentwicklung steht. Wenn eine Variable *summe* deklariert wird, ist allein in dieser Benennung eine Rolle (also auch Verhaltensmuster) definiert: Die Variable wird die *Summe* irgendwelcher Einzelwerte speichern (und nicht etwa ihr Produkt).

Sajaniemi [Sa02] hat die Rollen untersucht, die Variablen in Programmbeispielen in Informatik-Lehrbüchern spielen. Er beschreibt allgemeine Muster für Rollen wie z.B. *Konstanten*, die sich nicht ändern, oder *Stepper*, deren Inhalt in Iterationen um einen konstanten Betrag erhöht oder erniedrigt werden.

Im traditionellen Software-Engineering wird spätestens in der Entwurfsphase eine in sich schlüssige kohärente Welt, ein informatisches Modell (z.B. als Klassenstruktur) entwickelt, die als Grundlage für die Namenswahl für die einzelnen Elemente dient. Bei einer agilen Software-Entwicklung (z.B. Extreme Programming [B299]), entsteht der dramaturgische Kern des Projektes unter Umständen erst während des Prozesses. Man beginnt mit einer eher vagen „Projektmetapher“, die in einer einzigen Gestalt die Idee der Systemstruktur zum Ausdruck bringt. Erst im Verlauf des Projekts wird sie in Iterationen immer weiter entfaltet. Dabei kommt es vor, dass (im Rahmen eines Refactoring) Bezeichner des bereits funktionierenden Programms noch geändert werden, um die Begrifflichkeit konsistent zu halten. Dabei wird allein die literarische Qualität und nicht die algorithmischen Effizienz verbessert. Die Idee des Dramas als Interaktion

von Entitäten ist in der Objektorientierten Programmierung (OOP) kultiviert und formalisiert. Wer ein objektorientiertes Programm konzipiert, muss eine animierte Fantasiewelt eigenaktiver Entitäten erfinden.

- In einem objektorientierten Grafikprogramm wird eine Linie nicht mit Hilfe eines Stiftes verlängert sondern sie erhält den Auftrag, sich selbst länger zu machen. Sie ist (im Gegensatz zum realen Bleistiftstrich) eigenaktiv und wächst.
- In objektorientierten GUI-Implementierungen wird ein flüchtiges Ereignis wie z.B. ein Mausklick , zu einer dauerhaften Entität materialisiert, die von einem Eventhandler verarbeitet werden kann.

Die Beispiele zeigen, dass die Objekte eines Programms durch die beobachtbare Wirklichkeit zwar inspiriert, aber nicht von ihr einfach abgeleitet werden können. Programmieren ist Fiktion und nicht Dokumentation. Da die Objekte interagieren, gibt es eine Rollenverteilung und induzierte Handlungsabläufe wie in der griechischen Tragödie. Die Programmiersprache erlaubt, einer Klasse praktisch beliebige Operationen zuzuordnen. Die Kunst liegt darin, *sinnvolle* Konglomerate von Aktivität und Daten zu erschaffen. Ziel ist eine Entlastung des Arbeitsgedächtnis [De08]. Die ist dann gegeben, wenn eine Klasse vertraute Ganzheiten, – Gestalten im Sinne der Gestaltpsychologie [We25] repräsentiert. Dann kann man auch ohne Detailkenntnisse der Klassendefinition errahnen, welche Attribute und Methoden Objekte einer Klasse besitzen und welche nicht, und für welche Zwecke man sie verwenden kann.

2 Das Design der Studie

Dramatisieren ist eine Facette des Programmierens. Wer ein Computerprogramm entwickelt, braucht die Fähigkeit kleine, schlüssige Geschichten (genauer: Muster für Geschichten) zu erfinden, die Problemlösungen repräsentieren. Dazu gehört sprachliche Fantasie. Denn es müssen Metaphern und Dekorationen erfunden werden. Im Unterschied zu naturwissenschaftlichen Modellen, die ein strukturtreues Abbild eines Wirklichkeitsausschnittes liefern, sind informatische Modelle in ihrer Struktur durch die Wirklichkeit bestenfalls angeregt. Das wichtigste Qualitätsmerkmal eines informatischen Modells ist kognitive Beherrschbarkeit. Für den Informatikunterricht ergeben sich einige Fragen: Wie gut können Schülerinnen und Schüler metaphorisch formulierte Algorithmen verstehen? Können sie sie besser oder schlechter nachvollziehen als sachlich formulierte umgangssprachliche Algorithmen? Inwieweit kann man Schülerinnen und Schüler durch Metaphern (in Wort und Bild) inspirieren, eigenständig einen Algorithmus zu formulieren? Werden bestimmte Metaphern gegenüber anderen bevorzugt?

In den Jahren 2010 und 2011 habe ich mit insgesamt 119 Schülerinnen und Schülern aus den Jahrgangsstufen 9, 10 und 11 einer Gesamtschule Workshops zur naiven Algorithmik durchgeführt. Die Schüler hatten keine besonderen

Programmiervorkenntnisse. Nur einige wenige hatten Informatikunterricht, der sich aber nur sporadisch mit Programmierung beschäftigte.

Es gab zwei etwas unterschiedliche Typen von Workshops (W1 und W2) mit folgenden Teilnehmerzahlen:

W1: 58 Schüler, davon 28 m., 30 w., Durchschnittsalter 15,4 Jahre ($\sigma = 0,6$)

W2: 61 Schüler, davon 21 m., 40 w., Durchschnittsalter 15,4 Jahre ($\sigma = 1,1$)

Sie dauerten jeweils etwa 45 min und gliederten sich in drei Phasen: 1) Interpretation von vorgegebenen Algorithmen. 2) Entwicklung eigener Algorithmen 3) Test der eigenen Algorithmen. Ich beginne mit der Darstellung des ersten Typs W1 und beschreibe dann, anschließend, was beim zweiten Typ anders war.

In Phase 1 erhielten die Schüler ein Aufgabenblatt das zwei dramatisierte und zwei nicht-dramatisierte Algorithmen mit Bildern wie in Abbildung 1 enthält. Die Teilnehmer führten die Algorithmen jeweils aus, suchten das Lösungswort (eine sinnlose Buchstabenkombination) und schrieben es auf. Zum Schluss markierten sie die schwierigste und die leichteste Aufgabe.

In Phase 2 sollten die Teilnehmer selbst Algorithmen mit Metaphern entwickeln und konnten sich dazu aus einem Reservoir von insgesamt zehn Aufgabenblättern bedienen.

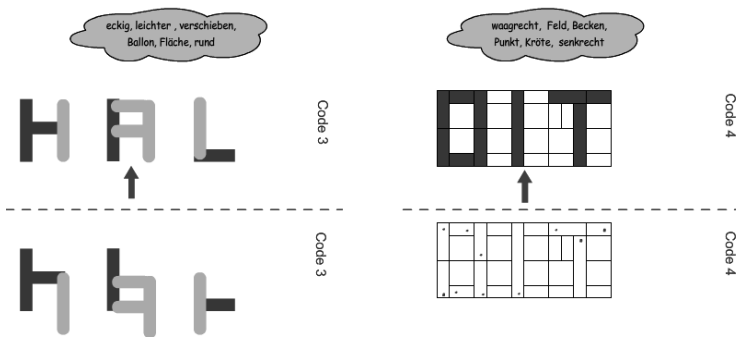


Abbildung 2: Eine Anordnung von Formen, die durch einen Algorithmus so verändert wird, dass drei Buchstaben entstehen.

Abbildung 2 zeigt zwei Beispiele. Auf dem Blatt sieht man zwei Bilder. Der Algorithmus beschreibt, durch welche Veränderungen man das untere Bild in das obere verwandeln kann. Dabei sollen wieder (wie in Phase 1) Buchstabenkombinationen entstehen. Die Probleme sind so aufgebaut, dass sie durch einen Algorithmus folgenden Typs gelöst werden: Eine einfache Aktivität (z.B. etwas verschieben oder eine Fläche ausfüllen) wird auf eine bestimmte Auswahl von grafischen Elementen angewendet. Es müssen also Kollektionen aufgrund gemeinsamer Merkmale definiert und Iterationen über diese Kollektionen ausgeführt werden. Die Probleme waren so gestellt, dass eine

Formulierung ohne Verwendung einer Iteration der Art „Mache mit allen ... folgendes...“ lang und komplex würde.

Die Teilnehmer der Workshops mussten also zuerst durch Induktion aus den gegebenen Beispiel-Transformationen ein allgemeines Verfahren finden und dieses dann explizieren. Man beachte, dass dies zwei unterschiedliche Leistungen sind. Ein Aktivitätsmuster kann auch quasi direkt als „prozedurales Wissen“ gespeichert und verwendet werden ohne expliziert zu werden [BB84] [An07]. Die Algorithmen sollten so formuliert werden, dass sie jeder der gleichen Altersgruppe versteht. Sie sollten in Phase 3 des Workshops getestet werden. Ein wichtiges Detail des Aufgabenblattes ist eine „Ideenwolke“ mit Begriffen, die eventuell für die Formulierung des Algorithmus verwendet werden können. Die Begriffe in der Wolken waren zum Teil metaphorisch (z.B. „Ballon“ für eine runde Form in Abbildung 2 links) und zum Teil sachlich beschreibend (z.B. „Fläche“). Den Teilnehmern war es freigestellt, diese Begriffe zu verwenden. Die Ideenwolken sollten nur helfen, Formulierungsschwierigkeiten zu überwinden. Die gesamte Teilnehmergruppe wurde in zwei Hälften A und B geteilt. Gruppe A erhielt andere Aufgaben als Gruppe B. Die beiden Gruppen durften in dieser Phase nicht miteinander kommunizieren. Am Ende trennten die Schüler an der gestrichelten Linie in der Mitte den oberen Teil des Blattes mit der Ideenwolke und der Grafik, die das Lösungswort zeigt, ab. Der untere Teil enthält dann nur den Algorithmus und die Ausgangssituation. .

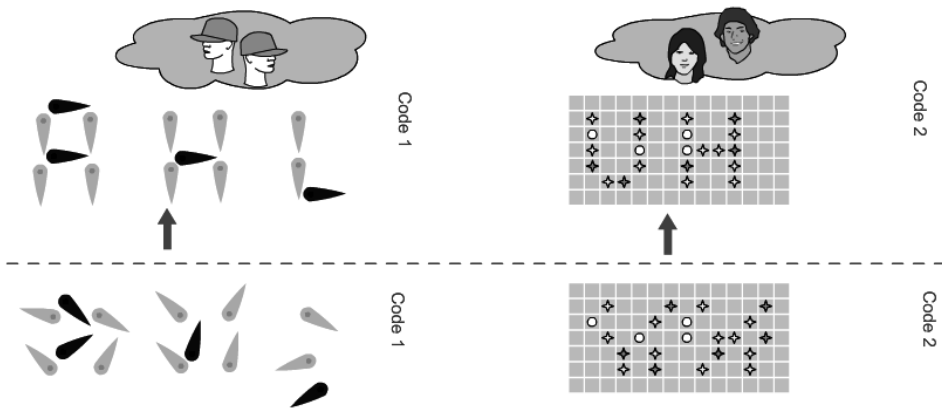


Abbildung 3: Zwei Aufgaben mit grafischer „Ideenwolke“.

In Phase 3 suchte sich jeder Schüler der Gruppe A einen Mitschüler aus Gruppe B. Die beiden tauschten Algorithmen aus und testeten, ob die Anleitung des Partners so verständlich war, dass das Lösungswort gefunden werden konnte.

Die Workshops des zweiten Typs (W2) weichen in folgenden Punkten ab: (1) Die Ideenwolken enthalten ausschließlich Metaphern - manchmal aus unterschiedlichen Domänen, so dass man sie nicht kombinieren kann. In vier der insgesamt zehn „Ideenwolken“ sind Zeichnungen anstelle von Texten (siehe Abbildung 3). Sie stellen Begriffe dar, die als Metaphern verwendet werden können.

(2) Zu Beginn von Phase 2 gibt es zunächst eine kurze Reflektion über Metaphern. Die Schüler erhalten dann explizit den Auftrag, nach Möglichkeit bei den Algorithmen eine der Metaphern aus der Ideenwolke zu verwenden

3 Ergebnisse

3.1 Dramatisierte Algorithmen verstehen

Phase 1 war bei beiden Workshop-Typen gleich. Insgesamt 119 Schülerinnen und Schüler haben die Aufgaben bearbeitet und je zwei dramatisierte und zwei nichtdramatisierte Algorithmen im Stil des obigen Beispiels (Abb. 1) nachvollzogen. Beide Varianten wurden im Schnitt zu genau 80% korrekt gelöst. Ebenso zeigte die Markierung von Aufgaben als besonders schwierig oder besonders leicht bei beiden Varianten keinen signifikanten Unterschied. Die metaphorische Ausdrucksweise war offenbar weder ein Hindernis (was aufgrund der längeren Texte hätte sein können) noch ein Vorteil.

3.2 Verwendung von Metaphern

Bei der Analyse der Algorithmen, die die Schüler selbst entwickelt hatten, wurden nur richtige oder fast richtige Algorithmen berücksichtigt.

In der ersten Workshop-Variante W1 war es freigestellt, Begriffe aus der Ideenwolke zu verwenden. Zudem waren nur einige Begriffe metaphorisch und andere beschrieben sachlich die Geometrie der Abbildung. Die Schüler hatten also völlig Freiheit in ihrer Ausdrucksweise. Im zweiten Workshop W2 wurden Schülerinnen und Schüler explizit aufgefordert, wenigstens eine Metapher aus der Ideenwolke zu verwenden. Beispiel: Zum Problem „Code 1“ (Abbildung 3 links) formulierte eine 15-jährige Schülerin in W2 folgenden Algorithmus:

„Es stehen viele Leute mit verschiedenfarbigen Kappen in jeweils drei Gruppen. Zeichne die Leute so ein, dass die Leute mit den helleren Kappen nach unten gucken und die Leute mit den dunkleren Kappen nach rechts gucken.“

Das ist ein Beispiel für einen dramatisierten Algorithmus auf der Basis einer Metapher, die in der Ideenwolke angeboten wurde. In W2 enthielten 8 von 11 überwiegend korrekten Lösung (8/11) diese Metapher. Das heißt: Fast jeder, der überhaupt in der Lage war einen passenden Algorithmus zu formulieren, hat auch die Kappen-Metapher verwendet. In W1 dagegen wurden nicht ein einziges Mal die in der Ideenwolke angebotenen Begriffe „Kappe“ oder „anschauen“ (die die gleiche Metapher unterstützen) verwendet.

In den 132 überwiegend korrekten Algorithmen aus W1 gab es insgesamt 50 Metaphern. Die 138 überwiegend korrekten Lösungen aus W2 enthielten dagegen insgesamt 286 Metaphern. Das heißt – etwas pointiert – die Schülerinnen und Schüler dieser

Altersgruppe sind durchaus in der Lage Algorithmen mit Metaphern zu formulieren, entscheiden sich aber dagegen, wenn sie die Wahl haben.

3.3 Dramatisierungen

Die 138 überwiegend korrekten Algorithmen aus W2 wurden an Hand festgelegter Kriterien pauschal dahingehend beurteilt, ob sie dramatisiert waren oder nicht. Im Prinzip sollten zwei Metaphern und eine Handlung erkennbar sein. Um die Validität des Bewertungsverfahrens zu testen, hatte eine Gruppe von acht Personen 10% der überwiegend korrekten Algorithmen auf der Basis der Kriterien und vier Prototypen beurteilt. Es ergab sich eine Übereinstimmung von 80%. Die folgenden zwei Beispiele gehören zu den Prototypen. Eine 15-jährige Schülerin schrieb als Lösung zum Problem „Code 2“ (Abbildung 3 rechts): „Die weißen Punkte sind weibliche Personen und die schwarzen Sterne sind Männer. Von links kommt ein starker Wind und bläst alle weißen Punkte ein Stück nach rechts. Auf einmal kommt Wind von rechts und bläst alle schwarzen Sterne nach links.“ Dieser Algorithmus wurde nicht als Dramatisierung gewertet. Denn die beiden Metaphern für die Sterne, die zunächst eingeführt worden sind, werden im zweiten Teil nicht verwendet um die Aktivität zu beschreiben. Dagegen ist der folgende Algorithmus einer 18-jährige Schülerin zum selben Problem dramatisiert: „Die blassen Sternchenchinesen gehen immer einen Schritt nach links während die temperamentvollen Dunkelhäutigen schon zwei Schritte nach links machen.“ Hier gibt es drei Metaphern, die in einem sinnvollen Zusammenhang stehen. Die Bewegung wird als „Schritte gehen“ beschrieben und das passt zu den Figuren, die als Metaphern für Kreisflächen und Sterne verwendet wurden. Während es in W1 fast keine Dramatisierungen gab, waren in W2 65 der 138 überwiegend korrekten Algorithmen dramatisiert. Das heißt die Teilnehmer kamen nicht von alleine auf die Idee, die Transformation der Bilder durch ein Drama zu beschreiben. Gleichwohl wirkte der Zwang zur Dramatisierung in W2 nicht als Bremse. In beiden Workshops lieferte jeder Teilnehmer im Schnitt 2,3 überwiegend korrekte Algorithmen ab.

Der Anteil der Dramatisierungen war bei den verschiedenen Problemen durchaus unterschiedlich. Zum Problem „Code 2“ (siehe Abb. 3) gab es unter den 17 überwiegend korrekten Lösungen nur drei Dramen (18%). Dagegen waren bei „Code 3“ (siehe Abb. 2) 12 der 14 überwiegend korrekten Lösungen dramatisiert (86%). Die Algorithmen der Mädchen enthielten etwa genauso viel Dramatik wie die der Jungen.

Geschlecht	Personen- zahl	Alter (SD)	Überwiegend korrekte Alg.	Dramen	Anteil dramatisiert
Mädchen	40	15,5 (1,1)	107	48	49%
Jungen	21	15,1 (1,0)	31	14	45%

Tabelle 1: Dramatisierung und Geschlecht

3.4 Eigenaktivität versus Passivität

Ein zentrales Merkmal des objektorientierten Programmierparadigmas ist die Idee des eigenaktiven Objektes, das zwar Aufträge empfangen kann, aber dann *selbst* seinen Zustand ändert. Dem gegenüber steht die Vorstellung, dass ein Agent die Ursache für Veränderung ist und passive Entitäten verarbeitet (Material und Werkzeug). Die folgenden beiden Algorithmen aus WS 2 zum Problem „Code 4“ (siehe Abbildung 2 rechts, allerdings mit anderer Ideenwolke) beinhalten eigenaktive Objekte:

„Es ist Brunftzeit. Die Schwarzpunktkröten blasen sich auf um zu imponieren. [...]“ (Schüler, 14 J.) „Auf ein paar Beeten liegen kleine Samen. Die Samen werden mit Wasser gegossen. Als die Blume wächst, entwickeln sich Wurzeln, die über das ganze Beet wuchern.“ (Schülerin, 14 J.)

Der folgende Algorithmus (zum gleichen Problem) dagegen beschreibt die Veränderung des Bildes nur als Veränderung passiver Entitäten durch einen Agenten.

„Male die Baumstämme aus, die von einem Specht heimgesucht wurden.“ (Schülerin, 17 J.)

Die Vorstellung eigenaktiver Entitäten (im Stil der OOP), wurde – je nach Problem – in sehr unterschiedlichem Ausmaß angewendet. Es konnte bei den untersuchten Beispielen keine generelle Tendenz ausgemacht werden. Zum Beispiel wurde das Vergrößern eines Elementes in manchen Fällen bevorzugt als Wachsen oder Dickwerden (eigenaktiv) und in anderen Fällen bevorzugt als Aufblasen oder Strecken (Veränderung einer passiven Entität) beschrieben.

4 Pädagogische Implikationen

Die Klassenraumaktivitäten, die in dieser Studie zum Einsatz kamen, haben auch lehrreiche Aspekte. Diese sollen nun angesprochen werden. Offenbar können Fünfzehnjährige umgangssprachliche Algorithmen zu Problemen wie in den Workshops innerhalb von wenigen Minuten formulieren und mit der Hilfe anderer Menschen auf Korrektheit und Verständlichkeit testen. Die Besonderheit gegenüber dem Programmieren ist nicht also nur das größere Entwicklungstempo (Minuten statt Stunden) sondern auch die soziale Komponente. Denn der umgangssprachliche Algorithmus ist für ein menschliches Auditorium. In den Workshops konnte beobachtet werden, dass die Teilnehmer durchaus an der Wirkung ihrer literarischen Algorithmen interessiert waren. Wird der andere mich verstehen? Wie gefällt ihm meine Formulierung?

„Schnelle“ erlebnisorientierte Übungen dieser Art könnten eine Ergänzung zu „langsamen“ konstruktiven Programmierprojekten sein. Denn bei der Entwicklung metaphorischer Algorithmen im Stil der Workshops praktizieren die Schüler „großes Programmieren im Kleinen“. Sie entwickeln plausible Systeme aus Akteuren und Aktivitäten. Genau das gleiche geschieht in der Anfangsphase einer

Softwareentwicklung, wenn eine Projektmetapher erfunden und eine Klassenstruktur designt wird.

Vorgegebene Metaphern können inspirierend sein und Kreativität provozieren. Ungewöhnliche Begriffskombinationen wie z.B. „Warze, aufblasen, Fliege, dick werden“ in der „Ideenwolke“ zu Aufgaben wurden von den Schülern als Herausforderung gesehen – nicht nur um einen Algorithmus zu erfinden sondern auch und vor allem ihn auf eigenwillige Art zu formulieren. Hier geht es nicht um eine produktbezogene *konstruktive* Kreativität wie sie z.B. Resnick mit Scratch hervorlocken will [Re07], sondern um *sprachliche* Kreativität. Auch sie ist (im Sinne Knuths) eine Facette der Programmierkunst.

Literaturverzeichnis

- [An07] Anderson, J. R.: Kognitive Psychologie. 6. Auflage. Spektrum Akademischer Verlag, 2007.
- [Ar69] Aristoteles: Über die Dichtkunst. Übersetzung von Friedrich Ueberweg. Berlin 1869.
- [Ba05] Baumgarten, H.: Compendium Rhetoricum. 2. Auflage. Vandenhoeck und Ruprecht, Göttingen 2005.
- [Be99] Beck, Kent (1999): Extreme Programming Explained. Boston u.a. (Addison Wesley).
- [BB84] Berry, D.C.; Broadbent, D.E.: On the relationship between task performance and associated verbalizable knowledge. Quarterly Journal of Experimental Psychology. 36A 1984; S. 209-231.
- [Cu06] Cummings, R. E.: Coding with power. Towards a Rhetoric of Computer Coding and Composition. In: Computers and Composition. Band 23 (4), 2006, S.430-443.
- [De08] Dehn, M.J.: Working Memory and Academic Learning. John Wiley & Sons, Hoboken, New Jersey 2008.
- [Fi02] Fishwick, P.A.: Aesthetic Programming: Crafting Personalized Software. In: Leonardo, 35/ 4, August 2002, S. 383-390.
- [He02] Herbst, Claudia (2002). Blood, sweat and code: A new text, power and illiteracy in the context of gender. The Journal of Literacy and Technology, 2 (1). Online (abgerufen am 27.1.2011) <http://www.literacyandtechnology.org/volume2/no1/herbst.html>
- [Kn84] Knuth, D. E.: Literate Programming. In: The Computer Journal 27/2, 1984, S. 97-111.
- [Re07] Resnick, M.: All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten. In: Proceedings of the 6th ACM SIGCHI conference on creativity & cognition, Wahington D.C., USA 2007, S. 1-6
- [Sa02] Sajaniemi, J.: Visualizing Roles of Variables to Novice Programmers. In: Kuljis, J.; Baldwin, L.; Scoble, R. (Hrsg.): Proc PPIG 14, Brunel University, 2002.
- [We25] Wertheimer, M.: Über Gestalttheorie. In: Philosophische Zeitschrift für Forschung und Aussprache, 1925, 1, S. 39-60.

Kompetenzorientierung und Schulrealität

Eine Skizze von Überlegungen im Spannungsfeld von
theoretischer Konzeption und praktischer Umsetzung

Peter K. Antonitsch

Institut für Informatiksysteme/Informatik Fachdidaktik
Alpen-Adria Universität Klagenfurt
Universitätsstraße 65-67
A – 9020 Klagenfurt
Peter.Antonitsch@uni-klu.ac.at

Abstract: Bildungsstandards für ein Unterrichtsfach definieren ein fachspezifisches Kompetenzmodell mit Kompetenzbereichen, die für praktizierende Lehrerinnen und Lehrer durch aus den Bildungsstandards abgeleitete Lehrpläne verbindlich werden. Diese Rahmenbedingungen werfen für die Praxis des kompetenzorientierten Unterrichtens eine Reihe von Fragen auf. Am Beispiel von Programmierunterricht im Kontext der in Entwicklung begriffenen kompetenzorientierten Lehrpläne für das berufsbildende Schulwesen in Österreich thematisiert der Artikel einige dieser Fragen und gibt mögliche erste Antworten.

1 Lernen und Kompetenz (und der Mensch)

Im Zusammenhang mit Bildungsstandards wurde *Kompetenz* zu einem vielstrapazierten Begriff. Das Erreichen von *Kompetenzen* aus wohldefinierten *Kompetenzbereichen* stellt das wünschenswerte Ergebnis von Lernprozessen in der Schule dar. Lernprozesse in der Schule sollen also idealerweise *kompetenzorientiert* ablaufen. Was bedeutet das?

H. Schwedes schreibt in [Sc05]: „Man hat etwas gelernt, wenn man etwas kann, das man vorher nicht gekonnt hat, nicht gewusst hat. [...] ob jemand etwas gelernt hat, kann man erst sehen, wenn die neue Fähigkeit gezeigt hat.“, und: „Zum effektiven Wissen gehört auch die Handlungsfähigkeit, in Situationen angemessen zu reagieren oder Probleme lösen zu können. Diese Kombination von Wissen und Können bezeichnen wir als Kompetenz.“

Diese Festlegungen sind noch zu ergänzen um: „Die Fähigkeit eines Menschen, bestimmte Aufgaben selbständig durchzuführen, wird als Kompetenz bezeichnet“ [HP05], bzw.: „Richtig verstandene Kompetenzformulierungen beschreiben eine neue Art von Fähigkeiten und beantworten die Frage, welche Fähigkeiten die Schülerinnen und Schüler besitzen müssen, um den heutigen Anforderungen gewachsen zu sein.“ [Pe05]

Das Neue an *Kompetenzorientierung* ist demnach nicht das Verknüpfen von Lernen und Kompetenzen, sondern

- das Festlegen von Denk- UND Handlungsmustern, die verbindlich erlernt werden sollen: Nur wer über adäquate Denk- und Handlungsmuster verfügt, kann diese anwenden, um in (neuen) Situationen angemessen zu reagieren. Und:
- der individuelle und selbsttätige Erwerb dieser Denk- und Handlungsmuster durch die Lernenden: Nur wer lernt, selbst tätig zu werden, kann Aufgaben selbstständig durchführen. Selbsttätigkeit benötigt aber auch die Möglichkeit der individuellen Annäherung und des individuellen (Lern-) Fortschritts. „Jeder Mensch ist einmalig. Und entsprechend einmalig gestaltet er vor diesem biografischen Hintergrund sein Lernen.“ [Mü03]

2. Kompetenzorientierung: Rahmenbedingungen und Fragen

Für Berufsbildende Höheren Schulen (BHS)¹ in Österreich liegt ein Kompetenzmodell für das Fach »Angewandte Informatik« vor, das den „Grundsätzen und Standards für die Informatik in der Schule“ [ABD08] nachempfunden zwischen der Handlungs- und Inhaltsdimension des Informatikunterrichts unterscheidet. Die Handlungsdimension kennt die Kompetenzbereiche »Verstehen« – »Anwenden« – »Analysieren« – »Entwickeln«, die Inhaltsdimension die Kompetenzbereiche »Informatiksysteme« – »Publikation und Kommunikation« – »Tabellenkalkulation« – »Datenbanken« – »Informationstechnologie, Mensch, Gesellschaft – Algorithmen, Objekte und Datenstrukturen«.²

Die Lehrpläne beinhalten neben einer Auflistung des »Lehrstoffs« auch als »Bildungs- und Lehraufgabe«(!) die Kompetenzformulierungen für die jeweiligen Kompetenzbereiche der Inhaltsdimension, z.B. für den im Folgenden interessierenden Kompetenzbereich »Algorithmen, Objekte und Datenstrukturen« des Faches »Angewandte Informatik«:

„Die Studierenden (!)

- können Ablaufalgorithmen entwerfen und Berechnungsschritte systematisch angeben;
- können Kommentare, Konstanten und Variablen in einer Programmiersprache darstellen und Befehlsstrukturen einer Programmiersprache anwenden;
- können die wichtigsten Datentypen unterscheiden, kennen ihre Einsatzbereiche und können Datenstrukturen und Objekte aus einfachen Datentypen zusammensetzen und komplexe Befehlsstrukturen erstellen.“.

Derartige Auflistungen von »Grobkompetenzen«, die von den Lernenden in Auseinandersetzung mit dem Programmieren erworben werden sollen, lassen für die praktische Umsetzung der Kompetenzorientierung einige Fragen offen, z.B.:

¹ »Berufsbildende Höhere Schulen« umfassen in Österreich technische, gewerbliche und kunstgewerbliche Schulen, kaufmännische Schulen, humanberufliche Schulen sowie Bildungsanstalten für Kindergartenpädagogik!

² In BHS mit »informatischem Schwerpunkt« (Abteilungen für Elektrotechnik, Elektronik oder Informatik an technischen Schulen) wird das Fach »Angewandte Informatik« ersetzt durch das Fach »Fachspezifische Softwaretechnik« oder »Fachspezifische Informationstechnik«, deren Kompetenzmodell sich vor allem in der Inhaltsdimension von dem der »Angewandten Informatik« unterscheidet. Der für das Folgende relevante Kompetenzbereich »Algorithmen, Objekte und Datenstrukturen« findet sich aber auch dort.

- Wie können Kompetenzen *operationalisiert* werden, sodass sie durch das Bearbeiten von Aufgaben individuell erworben werden können?
- Welche *Programmier-Lernumgebung* unterstützt die für sinnvoll und notwendig erachtete Individualisierung beim Kompetenzerwerb bestmöglich?
- Welche *Teilkompetenzen* können/müssen sinnvollerweise definiert werden, damit individueller Lernfortschritt sichtbar werden kann?
- Mit Hilfe welcher *Kompetenznachweise* kann der Erwerb von (Teil-) Kompetenzen beurteilt und bewertet werden?
- Wann ist im Laufe des Lernprozesses eine Kompetenz sinnvollerweise nachzuweisen?
- Wie kann erreicht werden, dass eine Kompetenz *längerfristig* verfügbar bleibt?

3. Aufgaben und Lernumgebungen

Nicht zufällig stehen die Fragen nach Aufgaben und Programmier-Lernumgebungen an erster Stelle: Aufgaben, die als Aufgaben wahrgenommen werden, fördern die Konzentration, „vermitteln Sachverhalte und Lösungspotentiale“ [Gi04, S.18]. Aufgaben sind damit im Unterricht schlechthin DIE Lerngelegenheiten, moderner ausgedrückt: DIE Gelegenheiten, Kompetenz zu entwickeln. »Aufgaben« werden aber individuell nur dann als sinnvolle (!) Aufgaben wahrgenommen, wenn sie eine Lücke zwischen dem Ist-Zustand und einem erwünschten Soll-Zustand darstellen [Gi04, S. 17]. Sinnvolle Aufgaben bedürfen also einer gewissen Vertrautheit mit der Aufgabensituation.

Im Informatikunterricht und besonders im Programmierunterricht wird die »Aufgabensituation« neben individuellen Vorerfahrungen der Lernenden auch durch die verwendete Software mitbestimmt, repräsentiert diese doch meist die erste sinnlich wahrnehmbare Programmiererfahrung. In dieser Ersterfahrung erleben die Lernenden, ob Aufgaben, die zu ihrer Lösung des Programmierens bedürfen, für sie »sinnvolle« Aufgaben sind. Nur wenn dies erfüllt ist, besteht auch die Bereitschaft, sich individuell mit der Aufgabe auseinanderzusetzen. Die erste These lautet daher: Die Wahl der Programmier- (Lern-) Umgebung und die in ihr lösbaren Aufgaben sind die zentralen Dreh- und Angelpunkte kompetenzorientierten Programmierunterrichts.

In [Ro09] werden Kriterien angegeben, die Software erfüllen sollte, die kreatives Lernen (und damit positive Lernerfahrungen) im Informatikunterricht, ermöglicht. Unter anderem soll die Software:

- intuitiv sein und geringe Einstiegshürden aufweisen,
- Prozessabläufe und Datenflüsse visualisieren,
- schrittweises und inkrementelles Lernen unterstützen,
- Beispiele und Ideen liefern.

Als Beispiele für Software im Programmierunterricht, die diese Kriterien erfüllen, werden Scratch, Alice oder Greenfoot genannt, die „das kreative Erstellen von Spielen, Animationen [...] in den Vordergrund stellen“ [Ro09]. Derartige Software stellt auch aus Sicht des Autors eine gute Option für kompetenzorientierten Programmierunterricht dar,

aus eigener Erfahrung aber folgt die zweite These: In Programmier-Lernumgebungen, die Kreativität in den Vordergrund stellen und damit für die Lernenden a priori sinnvolle Aufgaben ermöglichen, wird primär die Kompetenz entwickelt, mit den Strukturen »umzugehen«, die die spezifische Umgebung anbietet,

4. Praxiserfahrung: Selbsttätigkeit $\neq$ individueller Kompetenzerwerb

Die angesprochene »eigene Erfahrung« geht zurück auf das Schuljahr 2009/10, in dem der Autor für eine Lerngruppe³ ohne jegliche vorherige Programmiererfahrung unter dem Blickwinkel »Individualisierung« Scratch und Greenfoot (in dieser zeitlichen Reihenfolge) als Lernumgebungen für den Programmierunterricht auswählte (vgl. [An10] für eine detaillierte Darstellung). Die Schülerinnen und Schüler sollten unter anderem lernen, Aufgaben wahrzunehmen und zu lösen, die mit der verwendeten Software gelöst werden können, und grundlegende Programmierkonzepte anzuwenden.

Durch die Visualisierung der Prozessabläufe erleichtert Scratch das »Provozieren« von Lücken, sodass Aufgaben aus dem Lernprozess heraus von den Lernenden »gefunden« werden können. Dies und die einfache Anwendung der angebotenen Programmstrukturen durch eine intuitiv verständliche Repräsentation der »Programmiersprache« förderte bei Scratch die produktive Selbsttätigkeit (das »Lernen«?) der Schülerinnen und Schüler. Dadurch konnte bereits nach drei Monaten eine als Gruppenarbeit angelegte komplexe Abschlussaufgabe, die Simulation einer vierarmigen Verkehrskreuzung, von allen Schülerinnen und Schülern erfolgreich gelöst werden (vgl. Abbildung 1).

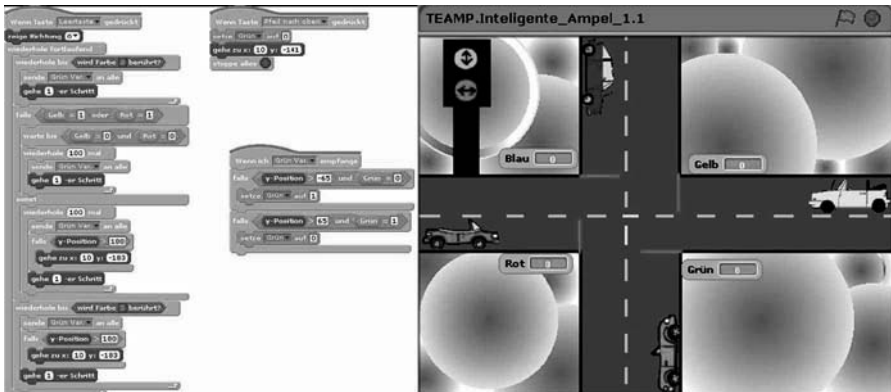


Abbildung 1: Lösung einer Gruppe zur Aufgabenstellung »Simulation einer vierarmigen Kreuzung. Der dargestellte »Programmcode« steuert nur eines der fünf Objekte!

Damit schienen die zuvor formulierten Lernziele erreicht: Die Lernenden konnten anspruchsvolle Aufgaben in der Programmier-Lernumgebung Scratch lösen, hatten also scheinbar (!) die Kompetenz entwickelt, grundlegende Programmierkonzepte anzuwen-

³ Die Lerngruppe bestand aus 16 Schülerinnen und Schülern in einem 1. Jahrgang (d.h. im Alter von 15 bis 16 Jahren) einer Höheren Technischen Lehranstalt.

den. Nach dem Wechsel zu Greenfoot⁴ zeigte sich aber, dass bereits das (textuelle) Programmieren einfacher Kontrollstrukturen den meisten Schülerinnen und Schülern erhebliche Schwierigkeiten bereitet. Während dies vordergründig mit der ungewohnten neuen Programmierumgebung begründet wurde, lag die eigentliche Ursache wohl eher darin, dass etliche der Lernenden für die in Scratch kennengelernten und bei der Lösung von Aufgaben mehrfach wiederholten Strukturen keine mentalen Modelle entwickelt hatten, die auf ähnlich strukturierte Systeme transferiert werden können.

Zwar zeigt die neurobiologische Forschung, dass „das Allgemeine dadurch gelernt [wird], dass wir Beispiele verarbeiten und aus diesen Beispielen die Regeln selbst produzieren“ [Sp02, S. 76], offenbar bedarf es aber flankierender Maßnahmen, dass spezifische Strukturen einer Programmierumgebung zu allgemeinen Programmierkonzepten abstrahiert werden können. Ohne diesen Prozess kann aber wohl nicht von Kompetenzerwerb gesprochen werden: Wie eingangs formuliert, ist Kompetenz nicht Können allein sondern die Kombination von Können UND Wissen.

Auch die Gleichsetzung der bloßen »produktiven Selbsttätigkeit« mit Individualisierung greift zu kurz, meint doch Individualisierung das Anstreben *gemeinsamer und verbindlicher Ziele (Kompetenzen)* auf individuellen Lernpfaden. Für kompetenzorientierten – und daher auch individualisierten – Unterricht genügt es daher nicht, die zu erwerbenden Kompetenzen nur festzulegen, sie müssen den Lernenden auch als »verbindliche Ziele« kommuniziert werden.



	A1	A2	B1	B2
THEORIE UND GRUNDLEGENDE HANDHABUNG	Ich kenne die wichtigsten Bestandteile einer Computereinrichtung.	Ich kenne die wichtigsten Grundbegriffe wie Datenspeicher und weiss, wo PC's überall eingesetzt werden können.	Ich kenne die Teile eines PC's von internen Geräten wie Grafikkarte bis zu den meisten Peripheriegeräten wie USB-Sticks. Die wichtigsten Abkürzungen und Begriffe kann ich zuordnen.	Ich kenne viele Begriffe dem IT –Bereich, sodass die Texte einer Computerzeitschrift grösstenteils verstehe. I kann Peripheriegeräte v Drucker selber installieren
COMPUTERBENUTZUNG UND DATEIMANAGEMENT (Desktop, Arbeitsplatz)	Ich kann Programme starten, darin arbeiten, speichern, drucken und anschliessend den PC wieder herunterfahren.	Ich finde Dateien, die ich im Netz oder auf einem USB-Stick gespeichert habe, wieder und kann damit weiter arbeiten und diese auf verschiedene Weisen speichern (speichern unter).	Ich arbeite sicher und effektiv in der Desktopumgebung. Ich kann im Arbeitsplatz oder Explorer meine Dateien und Ordner verwalten (umbenennen, löschen, kopieren, verschieben usw.). Ich kann mit den Desktop-Icons und mit Fenstern arbeiten. Ich weiss, wie man die Suchfunktion benutzt.	Ich kenne mehrere Möglichkeiten, Dateien ; verwalten und den Desk einzurichten (Arbeitspl Explorer, Startmenu usw kann die Eigenschaften, Startleiste und Taskleiste ändern.

Abbildung 2: Beispiel für einen Kompetenzraster grundlegender informatischer Bildung (Auszug, Quelle: http://www.institut-beateberg.ch/xs_datn/Materialien/kompetenzraster.pdf; 28.01.2011)

Dazu kennt die moderne Pädagogik Kompetenzraster, die die Lernenden VOR Beginn des Lernprozesses darüber informieren, welche Inhalte erworben werden sollen. Für jeden dieser Inhalte werden Kompetenzstufen angegeben, die von den Lernenden nach

⁴ Der Wechsel der Lernumgebung war notwendig, weil »es Schulkultur ist«, dass die Schülerinnen und Schüler am Ende des 1. Jahrganges (auch) Ansätze textbasierten Programmierens beherrschen.

individuellem Vermögen angestrebt werden können. Ergänzt werden diese Kompetenzraster durch Kompetenz-Checklisten, in denen jede Kompetenz durch operationalisierte Teilkompetenzen erklärt wird, sodass die Lernenden ihren Lernfortschritt auch selbst einschätzen können (vgl. Abbildungen 2 und 3). Diese Werkzeuge stellen nach Einschätzung des Autors die oben angesprochenen notwendigen flankierenden Maßnahmen für kompetenzorientierten Unterricht dar.

	Ich kann:	Ich trainiere:	
1	Ich kann Nomen erkennen und sie in die Einzahl oder Mehrzahl setzen.	- Wortstark 5,S.190; Werkstattheft 5 S.45,54 - Wortstark 6,S.189u.S.190; - Werkstattheft 6 S.56 - CD- R zu Wortstark 5/6 - Freiraum 5/6 Karten 28,29,30	
2	Ich kann zusammengesetzte Nomen erkennen und bilden.	- Wortstark 5,S.190;Werkstattheft 5 S.72 - Werkstattheft 6 S.54	

Abbildung 3: Kompetenz-Checkliste aus Deutsch/Sekundarstufe 1 zu: »Ich kann Nomen, Verben und Adjektive unterscheiden und kurze, einfache Sätze bilden« (Auszug, Quelle: [MBS08])

5. Ein Ansatz für kompetenzorientierten Programmierunterricht

Im laufenden Schuljahr 2010/11 wird vom Autor auf Basis der beschriebenen Erfahrungen mit zwei Lerngruppen⁵ kompetenzorientierter Programmierunterricht gestaltet. Als Kompetenzbereich wurde in Anlehnung an die angesprochenen Lehrpläne formuliert: „Die Studierenden können mit Kontroll- und Datenstrukturen, die in der verwendeten Programmierungsumgebung verfügbar sind, sinnvolle Aufgaben lösen.“ Als Strukturierungselemente werden adaptierte Versionen von Kompetenzraster und Checklisten verwendet.

5.1 Kompetenzraster, Checklisten und die Programmierungsumgebung

Kompetenzraster und Checklisten sind im Programmierunterricht von der verwendeten Software mitbestimmt⁶. Die Wahl der Programmier-Lernumgebung ist daher die erste Entscheidung bei der Planung kompetenzorientierten Programmierunterrichts. Dabei ist zu berücksichtigen, dass „Wissen [und] Handlungskompetenz [...] immer auch an spezifische Kontexte gebunden [sind]“ [Sc05, S. 8]. Eine Programmier-Lernumgebung sollte daher auf Basis einer Programmiersprache unterschiedliche Programmierkontexte (d.h. »Szenarien« und vordefinierte Objekte/Methoden) zur Verfügung stellen können, damit die Lernenden bestimmte Programmierkonzepte nicht exklusiv mit einem Programmierkontext verknüpfen.

Greenfoot erfüllt dieses Kriterium ideal, aus Gründen der »Schulkultur« musste aber C# (express) gewählt werden. Dies ist explizit keine Programmier-LERNumgebung, jedoch

⁵ Die Lerngruppen bestehen aus jeweils 17 Schülerinnen und Schülern in 1. Jahrgängen einer Höheren Technischen Lehranstalt, wiederum ohne nennenswerte Programmier-Vorerfahrung.

⁶ Beispielsweise stellt „Ich kann Bewegungen eines Objekts programmieren“ mit Greenfoot oder Scratch eine grundlegende, mit einer Standard-Java IDE aber eine sehr fortgeschrittene Kompetenz dar!

können durch (von Lehrenden) selbst programmierte »Projektvorlagen« unterschiedliche Programmierkontexte bereitgestellt werden. Für den beschriebenen Unterricht wurden ein von der Schildkrötengraphik [Pa85] inspirierte »Graphikroboterkontext« und ein Kontext mit grundlegender GUI-Ein- und Ausgabe vorbereitet (vgl. Abbildung 4).

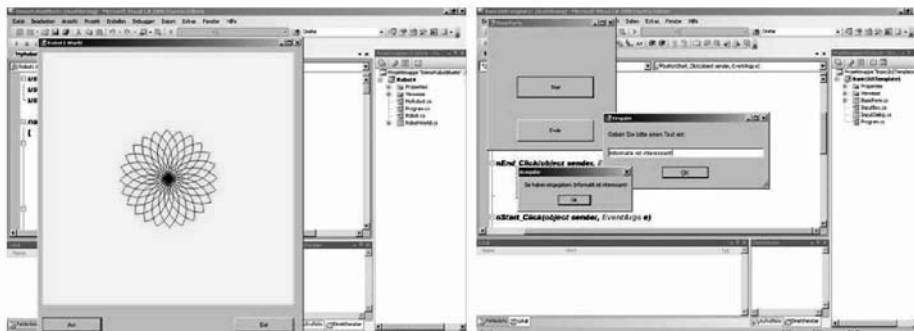


Abbildung 4: Die C#-Programmierkontexte »Graphikroboter« (links) und »BasicGUI« (rechts).

Auf Basis dieser Programmierkontexte wurden bislang⁷ die Kompetenzen:

- „Ich kann den »Graphikroboter« mit Hilfe eines Programms steuern.“ und
- „Ich kann Berechnungen, die dem Prinzip von Eingabe – Verarbeitung und Ausgabe folgen, sowohl programmieren als auch entsprechenden Code lesen und verstehen“

formuliert und durch Teilkompetenzen mit jeweils vier Kompetenzstufen konkretisiert. Für die »Graphikroboter«-Kompetenz sind dies z.B. die Teilkompetenzen:

- „Ich kann den »Graphikroboter« mit Hilfe von Schleifen und Methoden steuern.“
- „Ich kann den »Graphikroboter« mit Hilfe von Variablen, Schleifen und Methoden steuern.“
- „Ich kann den »Graphikroboter« mit Hilfe von Verzweigungen, Methoden mit und ohne Rückgabewert, Variablen und Schleifen steuern.“

Pro Kompetenzstufe werden den Lernenden vier Aufgaben angeboten (d.h. 16 pro Übungsblatt zu einer Teilkompetenz), um die Teilkompetenz auf der jeweiligen Kompetenzstufe erwerben zu können (vgl. Abbildungen 5 und 6).

Die Teilkompetenzen sollen zeitlich aufeinanderfolgend erworben werden, wobei die jeweils folgende Teilkompetenz auf der/den vorhergehenden aufbaut. Daher wurde bei der Festlegung der Kompetenzstufen darauf geachtet, dass jeweils auch Kenntnisse und Fertigkeiten der vorangegangenen Teilkompetenz in mindestens einer Kompetenzstufe »wiederholt« werden können. Hauptmotiv dafür ist, dass auch weniger leistungsfähige Schüler durch die Wiederholungen im Laufe des Lernprozesses die grundlegenden Teilkompetenzen (eventuell auf eher niedriger Kompetenzstufe) erwerben können.

⁷ Stand 31. Jänner 2011. Geplant ist noch das Anstreben der Kompetenzen (Arbeitstitel) „Bewusstes Verwenden von Objekten“ (im »BasicGUI«-Kontext) und „Programmsteuerung von Mindstorms-Robotern“ (im Kontext einer Java-Lejos Umgebung).

...zum Trainieren und Überprüfen der **1. Teilkompetenz**:

„Ich kann den »Graphikroboter« mit Hilfe von Schleifen und Methoden steuern.“

Kompetenzstufe			
D	C	B	A
Ich kann den Graphikroboter durch eine Folge von Befehlen in der act-Methode steuern.	Ich kann den Graphikroboter mit Hilfe von Schleifen in der act-Methode steuern.	Ich kann den Graphikroboter steuern, indem ich neue Befehle als Methoden programmiere und diese Methoden anwende.	Ich kann kompliziertere Bewegungsabläufe des Graphikroboters programmieren, indem ich selbst programmierte Methoden kombiniere.
Ich kann von Übungsblatt 1 die Aufgaben 1 bis 4 lösen.	Ich kann von Übungsblatt 1 die Aufgaben 5 bis 8 lösen.	Ich kann von Übungsblatt 1 die Aufgaben 9 bis 12 lösen.	Ich kann von Übungsblatt 1 die Aufgaben 13 bis 16 lösen.

Aufgabe 6:

Der »Graphikroboter« versteht auch den Befehl *Random(obergrenze)*, durch den er zufällig eine Zahl von 0 bis (inklusive) »obergrenze« berechnet. Wenn Sie also zufällig eine Zahl von 0 bis 10 »erzeugen« wollen und diese in einer Variablen *zufall* merken wollen, sieht die act-Methode so aus:

```
public void act()
{
    int zufall;
    zufall = Random(10);
}
```

- Steuern Sie den »Graphikroboter« mit Hilfe einer Schleife in der act-Methode so, dass er 1000 Schritte macht und sich nach jedem Schritt zufällig um einen Winkel von 0 bis 90° dreht.
- Überlegen Sie: Wie könnten Sie das Programm aus a) verändern, dass sich der »Graphikroboter« nach jedem Schritt zufällig um einen Winkel von -90° bis 90° dreht?

Abbildungen 5 und 6: Formulierung von Kompetenzstufen zu einer Teilkompetenz (D = niedrigste, A = höchste Kompetenzstufe) bzw. einer Aufgabe zur Kompetenzstufe C.

5.2 Kompetenznachweise und Leistungsbeurteilung

Die Unterteilung jeder Teilkompetenz in Kompetenzstufen hat einen pragmatischen Grund: Die Notenskala in Österreich sieht vier positive Beurteilungen vor. Da der Nachweis, dass eine Teilkompetenz (auf welcher Kompetenzstufe auch immer) erworben wurde, jedenfalls eine positive Leistung des Lernenden darstellt, helfen die vier gewählten Kompetenzstufen dabei, Kompetenznachweise auf Noten abzubilden.

A	18 P.	1	13,5 P.
B	14 P.	2	9 P.
C	10 P.	3	4,5 P.
D	6 P.	4	
	0 P.	5	

Abbildung 7: Kompetenzstufen (links) und Punkteverteilung bei »progressiven« Mitarbeitskontrollen. Mitte: Maximale Punkteanzahl für die beiden Aufgaben der jeweiligen Kompetenzstufe. Rechts: Abbildung der Gesamtpunkteanzahl auf das Notenschema.

Als Kompetenznachweise dienen einerseits die Lösungen der Übungsaufgaben, die die Schülerinnen und Schüler auf den für die Lerngruppen vorbereiteten moodle-Kurs hochladen können, andererseits erfolgt nach der Übungsphase zu jeder Teilkompetenz eine kurze schriftliche Mitarbeitskontrolle (ohne Computer) mit Aufgaben ähnlich jenen von den Übungsblättern. Da Individualisierung auch mit Selbsteinschätzung zu tun hat, werden diese Mitarbeitskontrollen in zwei Varianten durchgeführt:

- »*progressiv*«: Die Schülerinnen und Schüler erhalten ein Angabenblatt mit acht Aufgaben, zwei pro Kompetenzstufe. Nach einer Lesezeit von 10 Minuten muß – je nach Selbsteinschätzung – eine Kompetenzstufe gewählt werden und nur die zwei Aufgaben dieser Kompetenzstufe sind binnen 15 Minuten zu lösen. Da es aber hierbei zu Fehleinschätzungen der eigenen Fähigkeiten kommen kann, wurden die Aufgaben zusätzlich mit einem Punkteschema unterlegt (vgl. Abbildung 7).
- »*klassisch*«: Die Schülerinnen und Schüler erhalten ein Angabenblatt mit Aufgaben zu allen Kompetenzstufen, die je nach Kompetenzstufe verschieden viele Punkte »wert« sind. Nach einer Lesezeit von 10 Minuten können die Aufgaben beliebig binnen 15 bis 25 Minuten gelöst werden. Anders als bei der »progressiven« Variante ist zum Erreichen einer positiven Beurteilung hier aber eine Mindestpunktzahl > 0 notwendig.

6. Zwischenresümee und Ausblick

Die gemachten Erfahrungen mit kompetenzorientiertem Unterricht weisen auf neue(?) didaktische und pädagogische Betätigungsfelder hin, z.B.:

Kompetenzraster und Checklisten mit abgestimmten Übungsaufgaben helfen, kompetenzorientierten Unterricht transparenter zu strukturieren. Notwendig erscheint dabei, die allgemein gehaltenen Kompetenzformulierungen in Lehrplänen einerseits für die praktische Umsetzung zu detaillieren, andererseits durch Festlegung von Kompetenzstufen und/oder Teilkompetenzen auf die konkrete Lernsituation abzustimmen.

Besonderes Augenmerk ist darauf zu legen, dass die formulierten Kompetenzen für die Lernenden Bedeutung haben müssen. Zur Illustration diene ein bewusst einfach gehaltenes Negativbeispiel: Das Wissen darum, dass die Kompetenz „Ich kann eine Schleife programmieren.“ erworben werden soll, hat für den Lernenden vor dem Lernprozess keinen Informationswert, wenn er noch gar nicht weiß, was eine Schleife ist!

Im beschriebenen Ansatz wird diesem leicht zu übersehenden Problem dadurch begegnet, dass der »informatische Kern« der ersten beiden Kompetenzen identisch ist: Sowohl bei der Programmierung des »Graphikroboters« als auch beim Programmieren mit graphischen Benutzerschnittstellen liegt das Hauptaugenmerk auf den elementaren Strukturierungselementen Variable, Verzweigung, Schleife und Methode: Dadurch haben diejenigen Lernenden, die beim ersten »Kompetenzdurchgang« erst das notwendige Vokabular und eine vage Vorstellung dessen erworben haben, was »Programmieren« sein könnte, die Möglichkeit, beim zweiten »Kompetenzdurchgang« mit besseren individuellen Vorerfahrungen neu einzusteigen. Dies bedeutet eine Abkehr von dem Ansatz, sich mit (z.B.) Schleifen so lange zu beschäftigen, bis dieser »Inhalt« von allen Lernenden in allen Details verstanden wurde, hin zu dem Ansatz, kontextbezogen alle

Kontrollstrukturen wiederholt neu kennenzulernen, sodass alle Lernenden schrittweise ein »Bild« von der »Wirksamkeit« dieser Strukturen entwickeln.

Für die Leistungsbeurteilung sind angesichts standardisierter Bildungsziele im Kontrast mit individuell unterschiedlich schnellem Kompetenzerwerb der Lernenden neue Mechanismen zu überlegen. Kompetenznachweise auf Basis der Selbsteinschätzung der Lernenden erscheint ein möglicher Ansatz zu sein. In der beschriebenen Unterrichtssequenz war aber die Quote der Fehleinschätzungen noch recht hoch, weswegen der »progressiven« Variante die »klassische« zur Seite gestellt wurde. Notwendig ist wohl auch die Entwicklung einer *Kultur* der Selbsteinschätzung.

Bereits die beiden angesprochenen Punkte bergen einen nicht zu unterschätzenden Ideen- und Arbeitsaufwand. Zumindest die Weiterentwicklung und Optimierung der kollegialen Kooperation am Schulstandort ist damit eine weitere Gelingensbedingung für die Integration kompetenzorientierten Unterrichtens in die jeweilige Schulkultur.

Der Übergang von »herkömmlichem« zu kompetenzorientiertem (Informatik-) Unterricht ist nicht bloß ein Schritt, sondern ein längerfristiger Prozess, der des regen Austausches aller Beteiligten bedarf. Diese Erfahrungsskizze ist als Beitrag zu diesem notwendigen Austausch zu verstehen.

Literaturverzeichnis

- [AG10] Arbeitsgruppe „Bildungsstandards in Angewandter Informatik“: Das Kompetenzmodell (Version 1.18), 2010. http://www.bildungsstandards.berufsbildendeschulen.at/fileadmin/content/bbs/AGBroschueren/AngewInformatikBHS_V18_1_.pdf
- [AK08] Arbeitskreis Bildungsstandards: Grundsätze und Standards für die Informatik in der Schule. Beilage zu LOG IN, Heft Nr. 150/151, 28. Jg. (2008)
- [An10] Antonitsch P.: Erfahrungen zur Individualisierung im Programmierunterricht. In: Tagungsband zum Symposium „25 Jahre Schulinformatik“ in Melk, OCG, Wien 2010
- [Gi04] [Sich] Aufgaben stellen. Kallmeyersche Verlagsbuchhandlung, Seelze, 2004
- [HP05] Humbert L., Pasternak A.: Informatikkompetenzen in: LOG IN Heft Nr. 135, 25. Jg. (2005). http://www.sn.schule.de/~istandard/docs/bildungsstandards_2008.pdf
- [MBS08] Die neue Max Brauer Schule Hamburg, 2008.
http://www.maxbrauerschule.de/mbs/downloads/2008_neue_mbs_bsp.pdf
- [Mü03] Müller A.: Lernen ist eine Dauerbaustelle; 2003. http://www.institut-beatenberg.ch/xs_daten/Materialien/Artikel/artikel_lernen_als_dauerbaustelle.pdf
- [Pa85] Papert S.: Gedankenblitze. Rowohlt, Reinbek bei Hambur, 1985
- [Pe05] Penon J.: Kompetenzen, Standards, Selbstorganisiertes Lernen und Lernfelder: Modeerscheinungen oder neue Möglichkeiten? Vortrag an der Carl Ossietzky Universität Oldenburg, 2005. http://www.bics.be.schule.de/inf2/didaktik/vortrag_uni_oldenburg/kompetenz_begriff.html
- [Ro09] Romeike, R.: Softwaretools für kreatives Lernen im Informatikunterricht. In: Peters I.-R.: Informatische Bildung in Theorie und Praxis. Beiträge zur INFOS 2009, 13. GI-Fachtagung »Informatik und Schule«, S. 33ff; LOG IN Verlag Berlin, 2009
- [Sc05] Schwedes H.: Wie Kinder lernen; 2005. <http://www.gaebler.info/schwedes/lernen.pdf>
- [Sp02] Spitzer M.: Lernen. Gehirnforschung und die Schule des Lebens. Spektrum Akademischer Verlag, Heidelberg-Berlin; 2002

Alle Links wurden am 31. Jänner 2011 geprüft.

Fachdidaktisch begründete Auswahl von Informatiksystemen für den Unterrichtseinsatz

Dorothee Müller
Bergische Universität Wuppertal

Abstract: Die Probleme, die ein computerzentrierter Informatikunterricht mit sich bringt, sind ein unter verschiedenen Gesichtspunkten immer wieder diskutiertes Thema der Fachdidaktik Informatik. Dabei wird jedoch die Wahl des PCs als Informatiksystem nicht hinterfragt. Aber der PC ist im Informatikunterricht nicht ohne Alternativen, und die Frage, welche Auswirkung die Wahl eines anderen vollständigen und frei programmierbaren Informatiksystems haben kann, lohnt sich. In der Willy-Brandt-Gesamtschule in Bergkamen wird seit 2007 in der Oberstufe Informatikunterricht mit Mobiltelefonen als alleinigem Informatiksystem gestaltet. Wollen wir die Auswirkungen der Wahl eines vom herkömmlichen Computer abweichenden Informatiksystems untersuchen, sind vorrangig drei Bereiche zu betrachten: die Unterrichtsorganisation, das informatische Weltbild der Schülerinnen und Schüler und der Genderaspekt.

1 PCs im Unterricht

PCs werden im Unterricht immer präsenter. **In den verschiedensten Fächern** werden sie mit fachspezifischer Lernsoftware, mit Bürosoftware und für die Internetrecherche genutzt. Räume werden für entsprechende Unterrichtsstunden eingerichtet und in Schulbibliotheken und Schülerarbeitsräumen stehen den Schülerinnen und Schülern PCs für selbstständiges Arbeiten zur Verfügung.

Für den **Informatikunterricht** insbesondere gilt, dass PCs in einem Maße unterrichtsbestimmend sind, dass für viele Schüler die Begriffe »PC« oder »Computer« geradezu als Synonyme für das Fach stehen. Der Unterricht findet in Räumen statt, die bezeichnenderweise mal »Informatikräume« mal »Computerräume« genannt werden. Die physische Dominanz der Computer und die von ihnen diktierte starre Sitzordnung bringen Restriktionen der Unterrichts- und Sozialformen mit sich. In dieser Umgebung wird bei Schülerinnen und Schülern ein Bild vom Informatikunterricht und vom Fach Informatik geprägt, das einer Gleichsetzung mit Computerunterricht, oft sogar mit Anwendungsschulung nahe kommt. Auswirkungen dieser speziellen Fehlvorstellung auf den Genderaspekt des Informatikunterrichts sind nachweisbar (vgl. [SK08]).

2 Untersuchungsbereiche

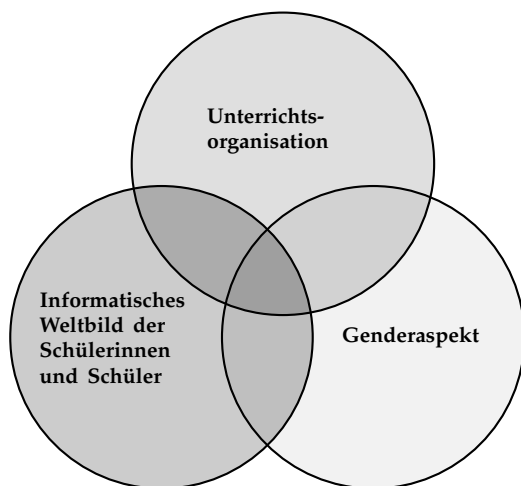


Abbildung 1: Untersuchungsbereiche

In Veröffentlichungen zur Fachdidaktik Informatik werden die durch die Dominanz von PCs im Informatikunterricht verursachten Probleme immer wieder und in verschiedenen Zusammenhängen beleuchtet. Die untersuchten Aspekte lassen sich drei, allerdings nicht scharf trennbaren, Bereichen zuordnen, die in Abbildung 1 dargestellt werden: der Unterrichtsorganisation, dem informatischen Weltbild der Schülerinnen und Schüler und dem Genderaspekt.

Verschiedene Konzepte zur »Informatik ohne Informatiksysteme« wie beispielsweise »Abenteuer Informatik«, »cs-unplugged« oder auch die Aufgaben des »Informatik Bibers« können als Grundlage für einen nicht an konkreten Informatiksystemen orientierten Unterricht gesehen werden. Die Materialien und Unterrichtsideen bereichern den Informatikunterricht, aber lösen das Problem des PC-dominierten Unterrichts nur teilweise. Informatiksysteme haben im Informatikunterricht wichtige Funktionen als Objekte des Erkenntnisinteresses und als Arbeitsmittel. Vor allem das aktive Realisieren eigener Modelle durch Implementieren ist ein wichtiger, kreativer und von dem zentralen Thema der informatischen Modellierung nicht trennbarer Teil der Schulinformatik. Hierzu sind vollständige, frei programmierbare Informatiksysteme notwendig.

Doch bedeutet dies nicht, dass PCs oder Laptops einen festen Platz im Informatikunterricht haben müssen. Es bieten sich Alternativen an. PCs oder Laptops sind zwar zur Zeit in praktisch allen Haushalten vorhandenen, doch sie stellen nur einen kleinen Teil der uns allzeit und allorts begleitenden Informatiksysteme und sind nicht die meistverbreitesten. Liegt es da nicht nahe, zu prüfen, ob an-

dere Informatiksysteme für die didaktischen Anforderungen des Informatikunterrichts geeigneter sind? Da es eine der Aufgaben des Informatikunterrichts ist, die Schülerinnen und Schüler zu einem verantwortlichen Umgang mit Informatiksystemen anzuleiten, sollte die Auswahl des benutzten Informatiksystems im Besonderen für den Informatikunterricht vorbildlich reflektiert sein.

3 Pilotprojekt Informatikunterricht mit dem Mobiltelefon

Zu der Frage, wie auf die Einschränkungen, die ein PC-dominierte Unterricht mit sich bringt, didaktisch reagiert werden kann, wurde von Humbert nach gemeinsamen Vorarbeiten mit Linkweiler [LH02] und Carrie [Ca06] ein konstruktiver Lösungsansatz entwickelt und an der Willy-Brandt-Gesamtschule in Bergkamen in einem besonderen Projekt realisiert: Ab dem Schuljahr 2007/2008 arbeitete dort ein kompletter Informatikgrundkurs der Oberstufe mit Mobiltelefonen als alleinigem Informatiksystem. Die Mobiltelefone werden als Unterrichtsgegenstand und als Arbeitsmittel zur Implementierung eigener informatischer Modelle genutzt. Dabei wird auf den Geräten der Quellcode (Python) direkt eingegeben. Und auch die Programme werden dort ausgeführt. In dem Projekt werden Geräte mit dem weit verbreiteten Betriebssystem Symbian S60 genutzt, wobei mit PythonForS60 und mit der PythonScriptShell und einem beliebigen Texteditor gearbeitet wird. Der Unterrichtseinsatz von Android-Geräten wäre jedoch ebenfalls möglich.

Erste Beobachtungen legten bald die Vermutung nahe, dass dieser Unterricht mit dem Informatiksystem Mobiltelefon eher geschlechtsunabhängig gestaltet werden konnte. Die Förderung durch Gleichstellungsmittel der Bergischen Universität Wuppertal ermöglichte es, im Schuljahr 2009/10 das Projekt auf eine breitere Basis zu stellen und zwei weitere Kurse mit Mobiltelefonen auszustatten. In der Unterrichtspraxis und durch begleitende informatikfachdidaktische Arbeiten¹ wurden Materialien für den Unterricht mit Mobiltelefonen erstellt, das Unterrichtskonzept sukzessiv verfeinert und einzelne fachliche und didaktische Aspekte des Projekts untersucht. Unter anderem wurden Erweiterungen der installierten Stifte-Mäuse-Bibliothek erstellt und so Möglichkeiten zur Visualisierung von Vektorgraphiken und zur Arbeit mit Datenbanken geschaffen.

Die Hypothesen zum Informatikunterricht mit dem Mobiltelefon, die während der Planung entwickelt wurden, schienen sich in der Projektpraxis zu bestätigen und wurden um weitere aus der Unterrichtspraxis gewonnene ergänzt. Unter anderem zeigte sich, dass offensichtliche Restriktionen, die mit Mobiltelefonen verbunden sind, wie kleines Display und unkomfortablere Eingabemöglichkeiten, bei den Schülerinnen und Schülern zu einer anderen Programmierpraxis führten. Statt des schnellen, zum Teil zufälligen, Erprobens vielfältiger Codevarianten auf der Suche nach einer richtigen Lösung, wurde eine stärkere Gewichtung auf das Modellieren gelegt. Zu untersuchen wäre, wie sich dies auf die Motivation und

¹Eine Auswahl der Beiträge: [HH08], [He09], [Lö10b], [Hu08], [Lö10a]

den Lernerfolg der Schülerinnen und Schüler auswirkt.

Im Jahr 2010 erreichte der erste Informatikkurs, der ausschließlich mit dem Informatiksystem Mobiltelefon unterrichtet wurde, das Abitur. Momentan wird das Projekt mit zwei Grundkursen der Oberstufe fortgeführt.

4 Zu untersuchende Hypothesen und Beobachtungen

Einsatzerfahrungen im Informatikunterricht wurden sowohl zu Laptops wie zu PDAs bereits dokumentiert. Jedoch gehören diese Geräte nicht der realen Lebenswelt der Schülerinnen und Schüler an, was hingegen auf das Mobiltelefon, wie später dargelegt wird, in besonderem Maße zutrifft. Da durch das Pilotprojekt grundlegend gezeigt wurde, dass es möglich ist, Mobiltelefone im Informatikunterricht in der gesamten gymnasialen Oberstufe als einziges Informatiksystem in der Hand der Schülerinnen und Schüler einzusetzen, bietet sich dies als Grundlage der Untersuchung an.

Zur Eignung des Mobiltelefons im Vergleich zum üblicherweise gewählten PC können im Rahmen der oben definierten Bereiche Beobachtungen beschrieben und Hypothesen aufgestellt werden. Die positiven Aspekte des Mobilfoneinsatzes im Informatikunterricht werden dabei betont, um gleichzeitig den Möglichkeitsraum, den das Pilotprojekt erschließt, aufzuzeigen.

4.1 Unterrichtsorganisation

Die Nutzung von PCs und Laptops im Informatikunterricht ist mit hohen Anschaffungs- und Wartungskosten verbunden und bindet Arbeitsressourcen der Informatiklehrerinnen und -lehrer. Der Unterricht findet in von PCs dominierten Räumen statt. Die Möglichkeiten, außerhalb des Unterrichts zu arbeiten, sind für die Schülerinnen und Schüler durch die Voraussetzung entsprechend konfigurierter Informatiksysteme eingeschränkt.

Finanzielle und personale Ressourcen: Die Anschaffungskosten für Mobiltelefone sind geringer als für PCs oder Laptops. Abhängig von dem Betriebssystem der Mobiltelefone können sogar vorhandene eigene Geräte der Schülerinnen und Schüler genutzt werden. Anders als die PCs der Schule können Mobiltelefone von den sie benutzenden Schülerinnen und Schülern selbst gewartet und für Unterrichtsaufgaben konfiguriert werden, so dass die finanziellen und personalen Ressourcen, die die Schule (bzw. der Träger) hierfür bereitstellen muss, entfallen.

Raum: Die Schülerplätze in herkömmlichen Informatikräumen können nur als Computerarbeitsplätze bezeichnet werden. Unterrichtsformen und Sozialformen sind in diesen starren Räumen vorgegeben, offene Unterrichtsformen sind nur schwierig durchführbar. Die Kommunikation innerhalb der Lerngruppe wird ge-

hemmt, die Unterrichtsgestaltung wird vom Arbeitsmittel bestimmt.

Für den Wechsel zwischen Unterrichtsphasen mit und ohne Informatiksystem müsste im Unterricht mit PCs sinnvollerweise ein zweiter Raum oder ein Raumteil mit Arbeitsplätzen ohne PCs zur Verfügung stehen. Allerdings kann wegen der damit verbundenen Unruhe und des Zeitaufwandes ein solcher Arbeitsplatzwechsel nur selten und für die gesamte Arbeitsgruppe gleichzeitig stattfinden. Die Bestimmungsmöglichkeit der eigenen Arbeitsphasen durch die Schülerinnen und Schüler ist reduziert.

Der Unterricht mit Mobiltelefonen kann hingegen in jedem Klassenraum durchgeführt werden. Ein spezieller Informatikraum muss nicht eingerichtet werden. Ein Arbeitsplatzwechsel für Arbeiten mit oder ohne Informatiksystem ist nicht notwendig. Dies bedeutet eine effektivere Nutzung der Unterrichtszeit.

Die Schülerinnen und Schüler können mit dem Mobiltelefon ihre Arbeiten mit nach Hause (oder an jeden anderen Ort) nehmen und dort im Rahmen von gestellten Aufgaben oder aus eigenem Interesse fortführen. Die Möglichkeit, das Programmieren zu üben, wird erhöht und Lernen aus Eigeninteresse wird gefördert.

Unterrichts- und Sozialformen: Die freie Wahl von Methoden und Sozialformen und offene Unterrichtsformen werden durch den Einsatz von Mobiltelefonen begünstigt. Die Mobilität des benutzten Informatiksystems und die Möglichkeit zur freien Einteilung der Arbeitsphasen fördert die Kommunikation zwischen den Schülerinnen und Schülern.

4.2 Das informatische Weltbild

Das »informatische Weltbild« der Schülerinnen und Schüler, im Sinne der subjektiven Konzeptualisierung des Faches Informatik (nach Schulte, Knobelsdorf [SK08]), wird von außerschulischen und schulischen Erfahrungen geprägt. Der ebenso häufigen wie falschen Gleichsetzung von Informatik und Computer müssen schulische Erfahrungen entgegengestellt werden. Der von PCs dominierte Informatikunterricht kann, indem er dieses falsche informatische Weltbild unterstützt, weder seiner wissenschaftspropädeutischen noch seiner allgemeinbildenden Aufgabe gerecht werden.

Wissenschaftspropädeutik: Dieser Unterricht versagt als Vorbereitung auf ein Informatikstudium, wie Romeike und Schwill in Langzeitbefragungen nachgewiesen haben. Sie kritisieren das »durch die Schule vermittelte Informatikbild, das vorwiegend auf Bedienungs- und Programmierfertigkeiten reduziert zu sein scheint« [RS06, S. 39]. Als eine der Konsequenzen »müssen offenbar viele Studierende ihre durch die Schule vermittelten Vorstellungen über Informatik im Studium schmerzhaft korrigieren und brechen das Studium daher frühzeitig ab« [RS06, S. 39]. Studierende mit dem Ziel Lehramt Informatik, die das Studium erfolgreich abschließen, ohne diese Fehlvorstellungen zu überwinden, geben durch

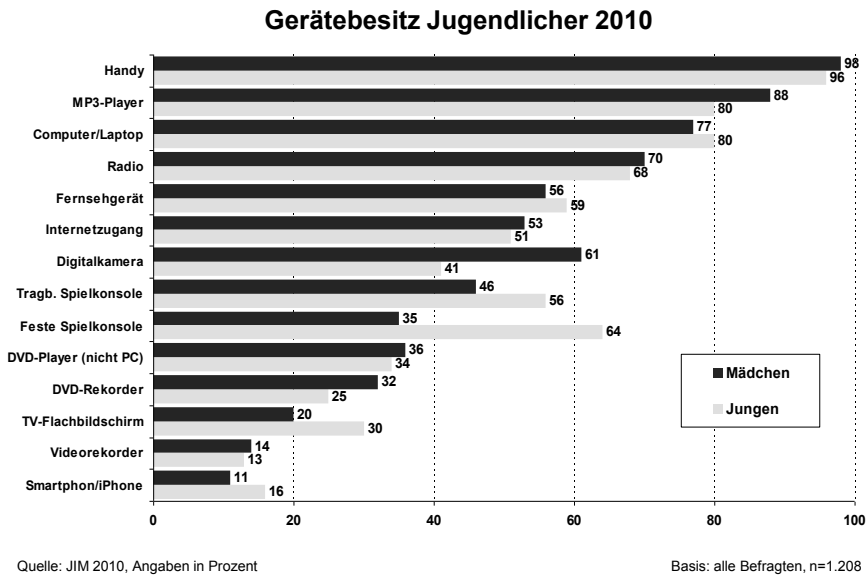


Abbildung 2: Gerätebesitz Informatikmittel lt. [MP10, S. 8]

fachlich geprägte Habitusformen dieses Informatikbild an die nächste Schülergeneration weiter (vgl. [HM10]). Wenn hingegen das Mobiltelefon als alleiniges Informatiksystem genutzt wird, werden Anwendungsschulungen verschiedener Büroprogramme nicht erwartet. Auch tritt die Tätigkeit des Programmierens als reine Code-Eingabe im Unterricht zugunsten des informatischen Modellierens in den Hintergrund.

Informatische Allgemeinbildung: Das informatische Weltbild der Schülerinnen und Schüler wird durch die Wahl des Informatiksystems PC unnötig reduziert. Unter der Vielzahl der uns allgegenwärtig umgebenden Informatiksysteme sind PCs keineswegs die häufigsten und auch nicht diejenigen, die die Lebenswelt der Schülerinnen und Schüler am stärksten bestimmen. Deutlich führt bei Jugendlichen von 12 bis 19 Jahren im Jahr 2010 das Mobiltelefon den Gerätebesitz an, wie die Abbildung 2 verdeutlicht.

Dies ist relativ unabhängig vom Alter der Jugendlichen: Bei einer Besitzrate von 95 Prozent kann man schon bei den 12- bis 13-Jährigen von einer Vollausrüstung sprechen (vgl. [MP10, S. 8]). Berücksichtigen wir weiterhin, dass die Jugendlichen ihr Mobiltelefon nahezu ständig mit sich führen und sie sich im Durchschnitt mit ihm häufiger als mit anderen Medien beschäftigen, so wird klar, dass Mobiltelefone die Informatiksysteme sind, die am stärksten zur Lebenswelt der Schülerinnen und Schüler gehören.

Dass die Tendenz zu einer noch vielfältigeren Mediennutzung über das Mobil-

telefon geht, verdeutlichen die Wachstumsraten innerhalb zweier Jahre: Bei der Nutzung von Mobiltelefonen gaben z. B. 4% der Jugendlichen 2008 und 13% 2010 an, das Mobiltelefon innerhalb der letzten 14 Tage für das Internet genutzt zu haben (vgl. [MP10, S. 26]).

Durch den Einsatz von Mobiltelefonen im Unterricht wird den Schülerinnen und Schülern in besonderem Maße bewusst, dass unsere Welt von Ideen und Artefakten der Informatik durchdrungen ist. Indem sie auf diesem, so eng mit ihrer täglichen Lebenswelt verbundenen Gerät, nach eigenen informatischen Modellen implementierend tätig werden, erfahren sie aktiv die Gestaltbarkeit der Welt durch die Informatik. Da sie außerdem selbst Administratoren ihrer Arbeitsgeräte sind, erleben sie sich nicht nur als Nutzer, sondern als Gestaltende und als Verantwortliche für das System.

4.3 Genderaspekt

Sollen im Informatikunterricht geschlechtsunabhängig gleichermaßen Schülerinnen wie Schüler erreicht werden, so lohnt sich bei der Wahl des Informatiksystems ein Blick auf die spezielle Lebenswelt von Mädchen. Hinsichtlich der Computer und Internetausstattung unterscheiden sich Mädchen und Jungen mittlerweile kaum noch. Jedoch ist die Nutzungshäufigkeit und -art unterschiedlich.

Medienpräferenz: Der herkömmliche Informatikunterricht regt auch »[...] motivierte Frauen nicht an, ein Informatikstudium zu beginnen« [RS06, S. 39]. Betrachtet man die Präferenzen der Jungen und Mädchen für bestimmte Informatiksysteme und ihre Einsatzmöglichkeiten, so wird ein Zusammenhang mit der Wahl des benutzten Informatiksystems wahrscheinlich.

Mehr Jungen als Mädchen beschäftigen sich täglich oder mehrmals pro Woche offline mit Computern (34% zu 28%), mit Computerspielen (55%) sogar fast viermal so viele Jungen wie Mädchen (14%). Mobiltelefone werden dagegen von etwas weniger Jungen als Mädchen täglich oder mehrmals pro Woche benutzt (88% zu 93%) – vgl. [MP10, S. 12]. Die Medienpräferenz für Mobiltelefone ist bei Jugendlichen geschlechtsunabhängig nahezu gleich hoch. Jedoch ist die Nutzungsvielfalt bei Mädchen höher: »Mädchen nutzen das Handy häufiger als Jungen für SMS, MMS, Fotos und Filme sowie als Kalender bzw. Terminplaner«, wird in [BI11, S. 13] festgestellt.

Selbstbild: Neben der Medienpräferenz prägt das Selbstbild bezüglich der eigenen informatischen Kenntnisse und Fähigkeiten den individuellen Zugang zur Informatik. Dieses Selbstbild wird stark von den persönlichen Computernutzungserfahrungen bestimmt. Mädchen schreiben sich hierbei vor allem Kompetenzen im Bereich der Anwenderfunktionen wie Textverarbeitung oder Fotobearbeitung zu, während Jungen sich eher Fähigkeiten zurechnen, die über die passive Nutzung hinausgehen, wie Programmieren und das Erstellen von Webseiten (vgl. [BI11, S. 17]). Schon 2008 stellen Schulte und Knobelsdorf fest: »In [...] Selbstver-

trauen und Vorkenntnisse in der Computernutzung – haben die Jungen deutliche Vorteile gegenüber den Mädchen« [SK08, S. 99]. Daraus schließen sie, dass im Informatikunterricht informatikfernen Personen, insbesondere Mädchen, zu einem »Konzeptwechsel«, der das kreative Entwerfen statt der »passive Nutzung« fokussiert, verholfen werden muss (vgl. [SK08, S. 102]). »Die beste Möglichkeit, hier zu intervenieren und damit (eventuell) den Frauenanteil in der Informatik zu erhöhen, wäre aus fachdidaktischer Sicht, einen Informatikunterricht anzubieten, der die Interessen, Wahrnehmungsmuster und Vorkenntnisse der Lernenden ernst nimmt« [SK08, S. 99].

Die Gleichsetzung von Informatiksystem und PC wird von Schulte und Knobelsdorf in diesem Zusammenhang nicht hinterfragt. Aber ihren ausgeführten Gedanken folgend, kann der Wechsel des benutzten Informatiksystems im Informatikunterricht bedeuten, der genannten Forderung nachzukommen und Interessen, Wahrnehmungsmuster und Vorkenntnisse der Lernenden ernst zu nehmen. Unter Berücksichtigung der hohen Kompetenz der Mädchen bezüglich des Informatiksystems Mobiltelefon, die nicht hinter der der Jungen zurücksteht, und der geschlechtsunabhängigen Affinität Jugendlicher für dieses Medium, ist zu erwarten, dass der Einsatz von Mobiltelefonen im Informatikunterricht auch den Schülerinnen zu einem besseren Zugang zur Informatik verhilft. In diesem Fall ist die individuelle Nutzungserfahrung nicht wie bei der Computernutzung für viele eine Barriere, sondern ein Einstieg zur Informatik.

Hinzu kommt, dass, abweichend vom herkömmlichen Unterricht mit PCs, nicht eine, meist männliche, Lehrperson oder eine größtenteils aus männlichen Schülern bestehende Informatik-AG die benutzten Informatiksysteme administrieren, sondern die Schülerinnen oder Schüler die volle, auch administrative Verantwortung für das Gerät übernehmen. Damit wird eine geschlechtsunabhängige Verantwortungspartizipation der Lernenden erreicht. Es ist zu untersuchen, ob die geschlechtsstereotypen Kompetenzzuschreibungen in der Informatik auf diese Weise abgebaut werden.

5 Fazit und Forschungsidee

Wenn der Informatikunterricht weiterhin zukunftsorientiert und mit der Lebenswelt der Schülerinnen und Schüler verbunden sein soll, muss auch das eingesetzte Informatiksystem auf seine didaktische Berechtigung geprüft werden. Eine in dem genannten Pilotprojekt erprobte Alternative stellt das Mobiltelefon dar.

Die geplante Forschungsarbeit auf der Grundlage dieses Projektes hat das Ziel, Hypothesen und Beobachtungen zur Auswirkung des gewählten Informatiksystems Mobiltelefon auf den Informatikunterricht zu evaluieren. Dabei wird die Übertragbarkeit der Untersuchungsfragen und -methoden für die Prüfung anderer Informatiksysteme angestrebt.

Datengrundlage: Bei der Planung eines Verfahrens mit dem Ziel, die Auswirkung

des gewählten Informatiksystems auf den Informatikunterricht zu untersuchen, scheint sich zunächst eine eng geführte Befragung der Lernenden und Lehrenden zweier Klassen oder Kurse, die sich nur im benutzten Informatiksystem unterscheiden, anzubieten. Doch dies kann bei der geringen Anzahl von möglichen Befragten nur zu Scheinergebnissen führen. Darüber hinaus ist es besonders bei den zu untersuchenden Bereichen »Informatisches Weltbild der Schülerinnen und Schüler« und »Genderaspekt« entscheidend, dass die Befragten selbst Zusammenhänge herstellen und die Relevanz setzen, so dass die Perspektive der Befragten gewahrt bleibt und eine offene Beschreibung der stets auch geschlechtsspezifischen Vorstellungen und Orientierungen ermöglicht wird. Daher ist ein offenes oder teilstandardisiertes Verfahren, wie z. B. das narrative Interview, das Leitfadeninterview oder das problemzentrierte Interview, zu wählen.

Auswertung: Zur Auswertung der Interviewtranskripte muss eine Methode gewählt werden, die es ermöglicht, nicht nur die explizit-reflektiven Vorstellungen der Interviewten, sondern auch eventuell nicht ausdrücklich formulierte Orientierungen zu Selbstbild, informatischem Weltbild und geschlechtsspezifischem Rollenbild zu rekonstruieren.

Literatur

- [BI11] BITKOM, Hrsg. *Jugend 2.0, Eine repräsentative Untersuchung zum Internetverhalten von 10- bis 18-jährigen*. Berlin, 2011. BITKOM – Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. http://www.bitkom.org/files/documents/BITKOM_Studie_Jugend_2.0.pdf – geprüft: 27. Januar 2011.
- [Ca06] Ralph Carrie. Einsatz mobiler Informatiksysteme im Informatikunterricht der gymnasialen Oberstufe. Hausarbeit gemäß OVP, Studienseminar für Lehrämter an Schulen – Seminar für das Lehramt für Gymnasien Gesamtschulen, Hamm, Juli 2006. <http://www.ham.nw.schule.de/pub/bscw.cgi/315319> – geprüft: 27. Januar 2011.
- [He09] Matthias Heming. Informatische Bildung mit Mobiltelefonen? Ein Forschungsbericht. In Bernhard Koerber, Hrsg., *Informatik und Schule – Zukunft braucht Herkunft – 25 Jahre INFOS – INFOS 2009 – 13. GI-Fachtagung 22.–24. September 2009, Berlin*, number P 156 in GI-Edition – Lecture Notes in Informatics – Proceedings, Seiten 134–145, Bonn, September 2009. Gesellschaft für Informatik, Köllen Druck + Verlag GmbH.
- [HH08] Matthias Heming und Ludger Humbert. Mobile Programming—the Usefulness of Mobile Phones for Teaching Informatics. In Roland T. Mittermeir und Maciej M. Syslo, Hrsg., *Informatics Education Contributing Across The Curriculum*, Seiten 54–63, Toruń, Poland, July 2008. Polish Information Processing Society. Presentation (M. Heming) <http://www.ham.nw.schule.de/pub/bscw.cgi/1104343> – geprüft: 27. Januar 2011.
- [HM10] Ludger Humbert und Dorothee Müller. Kultur und Informatiklehrerbildung. In *Informatik und Kultur – 4. Münsteraner Workshop zur Schulinformatik*, Seiten 78–87, 2010. Preprint:

<http://ham.nw.schule.de/pub/bscw.cgi/d1917318/HumbertMuellerMWS2010.pdf>
– geprüft: 10. Juni 2010.

- [Hu08] Ludger Humbert. Informatik und Gender – nimmt die Forschungsergebnisse ernst! In Marco Thomas und Michael Weigend, Hrsg., *Interesse wecken und Grundkenntnisse vermitteln – 3. Münsteraner Workshop zur Schulinformatik*, Seiten 81–90, Münster, Mai 2008. ZfL-Verlag. http://www.ham.nw.schule.de/pub/bscw.cgi/d1068247/2008-05-07_MWS-GenderErnstNehmen.pdf – geprüft: 27. Januar 2011.
- [LH02] Ingo Linkweiler und Ludger Humbert. Ergebnisse der Untersuchung zur Eignung einer Programmiersprache für die schnelle Softwareentwicklung – kann der Informatikunterricht davon profitieren? In Sigrid Schubert, Johannes Magenheimer, Peter Hubwieser und Torsten Brinda, Hrsg., *Forschungsbeiträge zur »Didaktik der Informatik« – Theorie, Praxis, Evaluation. 1. GI-Workshop DDI'02 (Schwerpunkt: Modellierung in der informatischen Bildung, 10.–11. Okt. 2002 in Witten-Bommerholz*, number P 22 in GI-Edition – Lecture Notes in Informatics – Proceedings, Seiten 119–128, Bonn, Oktober 2002. Gesellschaft für Informatik – Didaktik-Workshop, Köllen Druck + Verlag GmbH. http://www.die.informatik.uni-siegen.de/DIE_BIB/proceedings/gi-1ni-22.pdf – geprüft: 27. Januar 2011.
- [Lö10a] Susanne Löffler, Dorothee Müller, Janin Panske, Matthias Heming und Ludger Humbert. Artefakte und Genderladung – Konsequenzen für den Informatikunterricht? *magazIn – halbjährliches Magazin der Gleichstellungsbeauftragten der Bergischen Universität Wuppertal*, 4(Wintersemester 2010/11):29–34, Oktober 2010. <http://www.gleichstellung.uni-wuppertal.de/PDF/magazin/magazInWS1011.pdf> – geprüft: 30. Januar 2011.
- [Lö10b] Susanne Löffler. Von der objektorientierten Modellierung zur Datenbank – ein Konzept und seine Umsetzung mit Mobiltelefonen in der gymnasialen Oberstufe. Hausarbeit gemäß OVP, Studienseminar für Lehrämter an Schulen – Seminar für das Lehramt für Gymnasien Gesamtschulen, Hamm, Februar 2010.
- [MP10] MPFS. JIM 2010. Jugend, Information, (Multi-)Media. Basisstudie zum Medienumgang 12- bis 19-Jähriger in Deutschland. Forschungsbericht, mpfs, Stuttgart, November 2010. MPFS – Medienpädagogischer Forschungsverbund Südwest <http://www.mpfs.de/fileadmin/JIM-pdf10/JIM2010.pdf> – geprüft: 8. Dezember 2010.
- [RS06] Ralf Romeike und Andreas Schwill. Das Studium könnte zu schwierig für mich sein – Zwischenergebnisse einer Langzeitbefragung zur Studienwahl Informatik. In Peter Forbrig, Günter Siegel und Markus Schneider, Hrsg., *HDI*, number P-100 in GI-Edition – Lecture Notes in Informatics – Proceedings, Seiten 37–50, Bonn, Dezember 2006. Gesellschaft für Informatik, Köllen Druck + Verlag GmbH. <http://subs.emis.de/LNI/Proceedings/Proceedings100/GI-Proceedings-100-3.pdf> – geprüft: 27. Januar 2011.
- [SK08] Carsten Schulte und Maria Knobelsdorf. »Jungen können das eben besser« – Wie Computernutzungserfahrungen Vorstellungen über Informatik prägen. In Mechthild Koreuber, Hrsg., *Struktur und Geschlecht. Über Frauen und Männer, Mathematik und Informatik*, Baden-Baden, 2008. Nomos Verlagsgesellschaft.

PicoCrickets als Zugang zur Informatik in der Grundschule

Ralf Romeike

Dominik Reichert

Universität Potsdam
A.-Bebel-Str. 89
14482 Potsdam
romeike@cs.uni-potsdam.de

PH Schwäbisch Gmünd
Oberbetringer Straße 200
73525 Schwäbisch Gmünd
reichert.dr@googlemail.com

Abstract: Mit dem zunehmenden Einsatz neuer Medien in der Grundschule stellt sich die Frage nach Ansätzen für eine altersgemäße frühe informatische Bildung. Im diesem Beitrag werden PicoCrickets im Kontext von Robotik als gestalterischer Zugang zur Informatik diskutiert. Erste Erfahrungen des Einsatzes in der Grundschule werden dargestellt.

1 Einleitung

Spätestens seit der INFOS 2009 wird das Thema „Informatik in der Grundschule“ wieder verstärkt aufgegriffen¹. In mehreren Vorträgen wurden u.a. Möglichkeiten des Einsatzes von Soft- und Hardware in der Grundschule am Bsp. von Etoys [Fr09] oder dem OLPC bzw. einem „virtuellen Klassenzimmer“ [HH09] vorgestellt. Sprechen Herper und Hinz zwar noch davon, dass Kinder „schon in der frühkindlichen Erziehung an die Nutzung des Computers herangeführt werden“ können, zeigt ein Blick in die Praxis, dass Computer in der Grundschule längst Realität sind, allerdings weitgehend ohne informatikdidaktische Überlegungen eingesetzt werden und entsprechend kritisch reflektiert werden. „Informatik-technische Aspekte“ gelten in der Diskussion als „schwer vermittelbar bzw. nicht geeignet“ (z.B. [Hi04]), weshalb sich vielerorts der Computereinsatz auf die Benutzung von Office-Software, Internetnutzung und Anwendung von Lernsoftware beschränkt. Übersehen wird dabei, dass erst Grundkenntnisse der Informatik ein Verständnis der Wirkprinzipien von Informatiksystemen ermöglichen. Notwendig ist also weniger ein „Heranführen der Kinder an die Nutzung des Computers“ als ein Zugang, welcher einen Beitrag zur informatischen Bildung leistet und über Nutzungskompetenz hinausgeht. Die Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen der GI [GI00] fordern bereits für die Grundschule einen intuitiven, aber fachlich korrekten Umgang mit interaktiven Informatiksystemen. Versteht man das Gestalten von Soft- und Hardware als Ziel der Informatik, ist ein früher kreativer Zugang zu wählen. Neuentwicklungen im Bereich von Robotik und Pro-

¹ Viel zu lange wurde der Einsatz von Computern in der Grundschule ohne Berücksichtigung der Fachdidaktik Informatik diskutiert, z.B. in vielen Büchern zum Thema „Computer in der Grundschule“.

grammierungsumgebungen für Kinder versprechen einen frühen sinnvollen Einsatz von Computern, erste Erfahrungsberichte bestätigen dies (z.B. [Re07]).

PicoCrickets stellen einen Zugang dar, der an der Erfahrungswelt der Schüler anknüpft, insbesondere den als *Ubiquitous Computing* bezeichneten Trend zu allgegenwärtigen Computern. Schüler kommen nahezu täglich mit integrierten Computern in Berührung: Die Welt ist voller interaktiver Objekte. Wenn Kinder aber selbst interaktive Objekte erschaffen wollen, haben sie meist keine Idee, wie sie das machen können. Es entspricht dem Bildungsauftrag, ihnen einen innovativen gestalterischen Umgang mit solchen Systemen aufzuzeigen. PicoCrickets sind speziell für Kinder im Grundschulalter entwickelte Robotik-Kits, die es Kindern ermöglichen, z.B. interaktive Gegenstände oder Spielzeuge zu entwickeln. PicoCrickets wurden von uns hinsichtlich der Fragestellung, inwieweit sie sich als früher Zugang zur Informatik eignen, in einer Grundschule erprobt. Im folgenden Abschnitt erfolgt eine Betrachtung von Zugängen zur Informatik in der Grundschule. Anschließend werden PicoCrickets vorgestellt und eingeordnet sowie die Erfahrungen der Erprobung dargestellt und diskutiert.

2 Zugänge zur Informatik in der Grundschule

Mikrowelten

Umfangreiche Berichte zum Einsatz von Computern in der Grundschule gibt es bereits aus den 80er Jahren anhand der Programmiersprache LOGO (z.B. [HL84; Zi85]). Ziel war weniger das Vermitteln informatischer Konzepte, als ein Verständnis für mathematische und geometrische Strukturen im Sinne des konstruktionistischen Lernansatzes. Dieser hebt handelndes Lernen anhand von konkreten, persönlich bedeutsamen Gegenständen hervor: „*[Learning] happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe.*“ [PH91] Die Wissenskonstruktion geschieht demnach nicht nur im Kopf des Lernenden, sondern basierend auf einem Konstruktionsprozess in der realen oder virtuellen Welt. Hierbei entsteht ein Produkt, das ausprobiert, gezeigt, diskutiert, analysiert und auch bewundert werden kann. Der Lernende wird hierbei „eins“ mit dem betrachteten Phänomen, statt es nur „von außen“ zu betrachten. Mit LOGO wurde der Grundstein für die Entwicklung von Mikrowelten gelegt, die Lernumgebungen auf Computern bezeichnen, in welchen Lernerfahrungen „unbehindert von den Komplexitäten der Welt“ [Pa98] stattfinden können. So werden Mikrowelten zu „Brutwelten für Wissen“ [Pa82] und können als „Gewächshaus für eine bestimmte Spezies schlagkräftiger Ideen oder intellektueller Strukturen“ [Pa82] dienen. Schlagkräftige Ideen sollen einfach, allgemein, nützlich und ausgeglichen sein. Auch für das Erfassen informatischer Ideen und Konzepte eignen sich Mikrowelten: Basierend auf dem Vorbild LOGOs haben verschiedene Mikrowelten in den Informatikunterricht Einzug gehalten (z.B. Kara, Java-Hamster [RNH04; Bo05]). Dennoch werden Schwachpunkte dieser Mikrowelten kritisiert: So entspricht es einem fragwürdigen Informatikbild, wenn Programmierung und Informatik vor allem dazu dienen sollen, Marienkäfer oder Roboter durch Labyrinth zu bewegen um diverse Gegenstände aufzusammeln. Ebenso verletzen viele der Mikrowelten die LOGO zugrunde liegenden Ideen: Beim konstruktionistischen Lernen anhand einer Mikrowelt soll der Lernende in der Lage sein, ein persönlich be-

deutsames Produkt zu erschaffen. In der Regel handelt es sich bei den zugehörigen Aufgaben aber statt um die Erschaffung eines Produkts um das Finden einer Lösung zu artifiziellen *Problemlöseaufgaben* (vgl. [Ro08]). In solchen Kontexten kann konstruktivistisches Lernen nicht gelingen. PicoCrickets stellt eine Lernumgebung dar, die konstruktivistisches Lernen einer Mikrowelt mit der greifbaren Welt verbindet. Produkte sind auch haptisch erfahrbar, vorzeigbar und lassen sich in einem konkreten Einsatzfeld z.B. zum Spielen verwenden. Mit diesen Eigenschaften scheinen PicoCrickets gute Voraussetzungen mitzubringen, Kinder im Grundschulalter an Konzepte der Informatik heranzuführen.

Fundamentale Ideen der Informatik

Schwill [Sc01] analysiert auf Basis psychologischer Literatur zur Entwicklung von Kindern, welche kognitiven Voraussetzungen Kinder benötigen, um fundamentale Ideen der Informatik zu verstehen. Anhand von Beispielen wird gezeigt, dass Kinder im Grundschulalter in der Lage sind, sich mit informatischen Ideen, wie z.B. der strukturierten Zerlegung, der Hierarchisierung und algorithmischem Vorgehen umzugehen. Voraussetzung ist, dass „die Gegenstände [...] altersgemäß aufbereitet und im Unterricht unter Berücksichtigung der kognitiven Strukturen der Kinder und unterstützt durch Handlungen oder reale Gegenstände vermittelt“ werden.

Algorithmik in der Grundschule

Weigend [We09] präsentiert Ergebnisse einer Studie mit 120 Grundschulkindern im Alter von 8 bis 10 Jahren hinsichtlich deren Fähigkeit zur Umsetzung algorithmischer Texte. In Spielen mit „naiv algorithmischen“ Anweisungen konnte gezeigt werden, dass fast alle Schüler die Umsetzung der an Kontrollstrukturen einer Programmiersprache angelegten Anweisungen beherrschten. Als häufigster Fehlertyp zeigte sich das Ignorieren von Details, was den Kindern in einer interaktiven Umgebung vermutlich aufgefallen wäre.

3 PicoCrickets

PicoCrickets können als Neuentwicklung basierend auf den Erfahrungen mit Lego-Mindstorms-Robotern verstanden werden. Lego Mindstorms haben als konkrete Umsetzung der konstruktivistischen Idee im Informatikunterricht weite Verbreitung gefunden (z.B. [WB07]). Mit ihnen wird die Idee der Mikrowelten von der virtuellen auf die reale Welt übertragen. Kritisiert wird an Mindstorms-Robotern, dass die Konstruktionen schnell sehr komplex und damit für jüngere Kinder schwer erfassbar werden. [BD09] vermuten, basierend auf eigenen Erfahrungen, dass dieser Zugang noch nicht in vollem Umfang für Kinder im Alter von 10 Jahren geeignet ist. Auch wenn mit Lego Mindstorms sehr unterschiedliche Konstruktionen gebaut werden können, kommen meist nur Roboterfahrzeuge zum Einsatz (vgl. [WB07]). Eine tatsächliche Verankerung in der Erfahrungswelt der Kinder erfolgt damit, abgesehen von Spielzeugautos, nicht. Mit PicoCrickets wird die Idee programmierbarer Bausteine auf die Welt der allgegenwärtigen Computer ausgedehnt. PicoCrickets sind kleine programmierbare Bausteine, an welche Sensoren (Licht, Berührung, Widerstand, Geräusch) als Eingabequellen ange-

Programmierbausteine möglich. Da PicoBlocks auf LOGO basiert, ist auch die textuelle Programmierung mit LOGO möglich.

Informatische Bildung und PicoCrickets

Aus Sicht des Informatikunterrichts sind PicoCrickets aus folgenden Gründen interessant:

1. PicoCrickets stellen ein gestaltbares Informatiksystem dar.
2. PicoCrickets werden in einer einfachen, für Kinder reduzierten Programmiersprache programmiert. Die Vorteile einer Mikrowelt bleiben dabei erhalten.
3. Mit PicoCrickets lässt sich Kreativität im Informatikunterricht erfahren und umsetzen.
4. Informatikunterricht mit PicoCrickets kann im besonderen Maße den Leitlinien informatischer Bildung entsprechen: Wirkprinzipien von Informatiksystemen lassen sich z.B. über virtuelle Objekte hinaus erfahren.

Robotik wird in der Sekundarstufe bereits als motivierender Zugang zur Informatik eingesetzt. PicoCrickets ermöglichen zum einen den früheren, aber inhaltlich vergleichbaren Einsatz dieses erfolgreichen Ansatzes. Der Kontext ändert sich zwar, aber die zu erwerbenden Fachkompetenzen sind gleich. Auch wenn die Aktivitäten sich unterscheiden: In der Anwendung der Strategie unterscheidet es sich nicht wesentlich, ob ein Roboter, wenn er auf ein Hindernis trifft, die Richtung wechseln soll oder ob ein interaktives Stofftier, wenn es gestreichelt wird, einen Ton von sich gibt. Die Konzepte sind gleich, einen Unterschied macht nur der Kontext. Darüber hinaus bieten PicoCrickets die Chance, eine breitere Schülerschaft anzusprechen. [RRB08] verdeutlichen, dass es auch bei Robotik notwendig ist, Schülern vielfältige Zugänge zu ermöglichen: *“Different students are attracted to different types of robotics activities. Students interested in cars are likely to be motivated to create motorized vehicles, while students with interests in art or music are likely to be more motivated to create interactive sculptures.”* Informatikunterricht muss mit seinen Aktivitäten unterschiedliche Schülerinteressen berücksichtigen, nicht nur hinsichtlich der gestellten Aufgaben. Resnick [Re07] verdeutlicht die Auswirkungen der häufig anzutreffenden Einseitigkeit von Mindstorms-Robotik-Projekten: *“Even with strong efforts to increase female participation, only 30% of the participants in LEGO robotics competitions are girls. In Cricket activities at museums and after-school centers, participation has been much more balanced among boys and girls.”*

Der Einsatz facettenreicher Robotik hat das Potenzial, auf wesentlich umfangreichere Weise als gängige Roboter-Vorstellungen es vermitteln, zu verdeutlichen, wie Informatik das tägliche Leben durchdringt und z.B. in Alltagsgegenständen wiederzufinden ist. So können Kindern sensibilisiert werden, Computer im Alltag wahrzunehmen, kritisch zu hinterfragen und ggf. auch Grenzen der Möglichkeiten von Computern aufzudecken. Mit PicoCrickets erhalten Kinder die Möglichkeit, sich eigene Einsatzszenarien für Computer in ihrer Erfahrungswelt auszudenken und zu realisieren. Die Gegenstände werden dabei zu „Things That Think“ und im konstruktivistischen Sinn für die Kinder zu „Things To Think With“ (vgl. [RBM96]).

Kompetenzerwerb mit PicoCrickets

Die GI-Empfehlungen zu Bildungsstandards beschreiben Kompetenzen der Informatik, die Schüler ab der 5. Klasse erwerben sollen. PicoCrickets stellen ein Werkzeug für den Informatikunterricht zur Verfügung, das es bereits jüngeren Kindern ermöglicht, vorbereitende Kompetenzen v.a. in den Inhaltsbereichen Algorithmen, Informatiksysteme, Informatik und Gesellschaft sowie in allen Prozessbereichen zu erwerben. Beispielhafte Kompetenzen sind in Abb. 2 aufgeführt. Insbesondere im Bereich der Algorithmik ermöglichen PicoCrickets bereits jüngeren Kindern, komplexere Kompetenzen zu erwerben.

Inhaltsbereich	Kompetenz Die SuS
Algorithmen	entwerfen, implementieren und beurteilen Algorithmen
Informatiksysteme	ordnen Bestandteile eines Informatiksystems der Eingabe, der Verarbeitung und der Ausgabe zu.
Informatik und Gesellschaft	nennen und beschreiben Informatiksysteme in ihrer unmittelbaren Lebenswelt

Abb. 2: Beispiele für beim Lernen mit PicoCrickets zu erwerbende Kompetenzen.

Erfahrungen mit PicoCrickets

In der informatikdidaktischen Forschung sticht ein Untersuchungsgegenstand regelmäßig hervor: Versuche, dem geringen Interesse an der Informatik bei Studierenden und Schülern zu entgegenen, u.a. dadurch, dass die Diversität der Informatik für Schüler deutlich gemacht wird. Informatik wird von Schülern häufig als „Programmieren, Mathe und ein bisschen Hardware“ wahrgenommen [MW06] und als langweilig, unsozial und un-kreativ eingeschätzt [YB07]. Aus den Daten der KIM-Studien [MFS08] geht hervor, dass 2000 immerhin 8% der Kinder zwischen 6 und 13 Jahren am Computer programmierten, im Jahr 2008 ist es nur noch die Hälfte, trotz der Bemühungen um informatische Bildung, einfacheren Programmierungsumgebungen und größerer Verbreitung der Computertechnik.

Ein früher Kontakt mit gestalterischen Aspekten der Informatik mit PicoCrickets wird in verschiedenen Projekten als Ansatz verwendet, o.g. Problemen entgegenzuwirken. In [BBE09] wurden PicoCrickets erfolgreich eingesetzt, um vor allem Mädchen für Informatik zu begeistern. Ziel der mehrjährigen Workshopreihen ist der Versuch, die „computing education pipeline“ des gesamten Bundesstaates Georgia zu ändern, indem bereits in der Grundstufe (pre-teen level) ein Interesse für Informatik angeregt wird. In der Auswertung der Workshop-Reihen über mehrere Jahre stellt Guzdial [Gu10] fest, dass PicoCrickets gemeinsam mit Scratch den größten Effekt hinsichtlich einer zur Informatik positiven Einstellung hatten, während im Vergleich Lego-Mindstorms-Workshops nur geringe positive Auswirkungen zeigten. PicoCrickets wurden auch von [MKL10] eingesetzt, um Interesse an den MINT-Fächern vor allem bei Mädchen und unterrepräsentierten Minderheiten zu fördern. Beim Erstellen von interaktiven Kunstwerken wurde handlungsorientiert gelernt und Kreativität gefordert. Einstellungen und Enthusiasmus gegenüber Ingenieurwissenschaften und entsprechenden möglichen Berufen verbesserten sich signifikant. Von vergleichbaren Erfahrungen berichtet [Ha10].

Trotz der weiten Verbreitung von Lego Mindstorms in deutschen Schulen, dem Einsatz dieser Robotik-Kits im deutschen Informatikunterricht und entsprechend vielen Berichten hierüber gibt es noch keine dokumentierten Erfahrungen mit PicoCrickets im Informatikunterricht. Dies war ein Grund für die Autoren, in einem Unterrichtsversuch den schulischen Einsatz von PicoCrickets in der Grundschule zu erproben.

4 Erprobung

Ziel der Erprobung war es, erste Erfahrungen im Einsatz von PicoCrickets in der Grundschule zu sammeln und festzustellen, ob Schüler in der Lage sind, selbständig kleine Projekte umzusetzen und zu untersuchen, ob sich die zuvor geschilderten positiven Erfahrungen im Projektunterricht bestätigen. Methodisch lehnte sich der Unterricht an folgenden Strategien zur Erhöhung der Beteiligung an Robotik-Projekten an [RRB08]:

1. Focus on Themes (Not Just Challenges)
2. Combine Art and Engineering
3. Encourage Storytelling
4. Organize Exhibitions (Rather than Competitions)

Die Schüler wählten verschiedene umzusetzende Themen, die nicht im gegenseitigen Wettbewerb standen, erdachten sich eine kleine Geschichte zu ihren Kreationen und präsentierten diese anschließend in einer kleinen Ausstellung. Die Umsetzung wird im Folgenden anhand zweier Beispiele illustriert. Durchgeführt wurde die Erprobung in einer aus sechs Kindern bestehenden Kleingruppe einer 4. Klasse einer integrierten Grund- und Werksrealschule in insgesamt vier Unterrichtsstunden.²

Projekt Katapult

Der 10 Jahre alte Dennis war von der ersten Sekunde an von der Arbeit mit den PicoCrickets begeistert. In der ersten Stunde wollte er zusammen mit zwei Mädchen ein Katapult bauen. Das Katapult sollte feuern, sobald Dennis eine Taste drückt. Da ihm dies aber relativ schnell zu einfach wurde, brauchte er unbedingt einen zweiten Sensor, welcher den Motor ebenfalls startete. Er wählte dazu den Geräuschsensor aus und klatschte. Nun stand er noch vor dem Problem, wie denn der Motor des Wurfarmes, der an einem Hebel geblockt wurde, wieder ausgeschaltet werden kann, um den Antrieb nicht zu beschädigen. Ohne großen Aufwand entschied er sich für die einfachste aller Varianten - er schaltete das Programm über den Knopf am Hauptbaustein aus. Zu guter Letzt stand die Frage im Raum, was man denn mit einem solchen Katapult alles machen könne? Kurzer Hand wurde eine passende Wurfchale gebastelt und ein Stoffbällchen-Weitwurf-Wettkampf im Klassenzimmer veranstaltet.

² Zum Zeitpunkt des Versuchs standen nur wenige PicoCricket-Sets zu Verfügung.

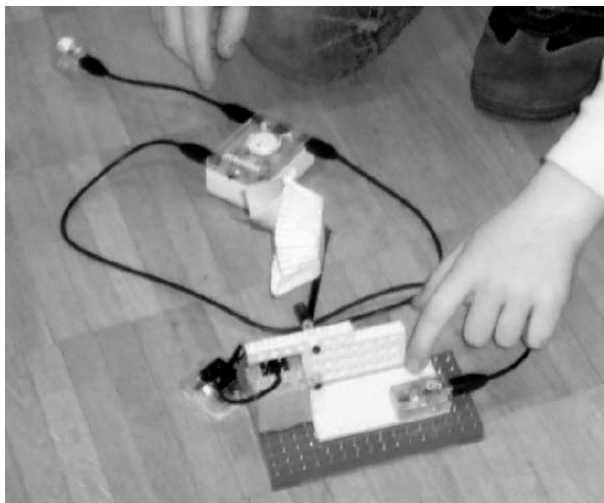


Abb. 3: Ein interaktives Katapult.

Projekt Widerstandsmessung

Nelly und Elisabeth, beide 10 Jahre alt, wollten ein Messgerät für den elektrischen Widerstand entwickeln und dieses anschließend testen. Als Hilfe nutzten die beiden eine den Baukästen mitgelieferte englischsprachige Anleitung. Schnell konnten sie das gewünschte Programm mit der passenden Apparatur entwickeln und erprobten dieses. Ein besonderes Highlight war für die beiden nicht nur die Anzeige, welcher Widerstandswert erreicht wurde, sondern die von ihnen als zweites Ausgabegerät installierte Lampe, die je nach Widerstand ihre Farbe wechselte. Besonders schön zu beobachten war dieser Effekt, wenn die Krokodilklemmen in Wasser hin- und herbewegt wurden. Schnell erkannten die Kinder anhand dieses Versuchs, dass Wasser Strom leitet und somit die Belehrung vom „Fön und der Badewanne“ ihre Richtigkeit besitzt. Die beiden befestigten ihr kleines Messgerät auf einem Lego-Brett und liefen damit durch das Klassenzimmer und suchten nach Leitern und Nichtleitern. Den Abschluss bildete eine Schlange aus 4-5 metallenen Gegenständen, die einen Stromkreis schlossen.

Weitere Projekte umfassten folgenden Themen: „Der Keinohrhasenhund BELLO“, „Weihnachten 2010“, „Weihnachtsglocken“, „Alien“, „Auto“, „Die Weihnachtskirche“ und „Eine süße Katze“.

5 Diskussion und Ausblick

Die Kinder zeigten alle ein großes Interesse an der Möglichkeit, eigene interaktive Objekte, die sie selbst vorher nur aus der Spielzeugabteilung kannten, zu bauen und zu programmieren. Die Erkundung der beiden zu Beginn mitgebrachten und vorgestellten Konstrukte sorgte für eine besondere Motivation. Der Umgang mit dem Computer stellte

für die Schüler kein Problem dar. Der Bau der Konstrukte war für die Kinder das Motivierendste an diesem Tag. Sie haben sehr interessiert an den Bauanleitungen gearbeitet und diese zunächst originalgetreu umgesetzt. Nach einer kurzen Anregung, dass die Bauanleitung keinen Endzustand darstellen muss, haben die Kinder selbständig mit weiteren Bauelementen und Sensoren gearbeitet. Hinsichtlich der Programmierung war anfangs Hilfestellung nötig, dann genügte es meist, wenn ihnen einmal die Vorgehensweise im Programm gezeigt wurde. Auch die Begrifflichkeiten prägten sich die Kinder sehr schnell ein. Am Ende jeder Einheit stand eine kurze Präsentation der Projekte mit einer Reflexion seitens der Kinder. Dies erlaubte den Kindern nochmals über ihre Erfahrungen, ihre Arbeit und ihren Lernprozess nachzudenken.

Bei beiden Projekten hatten die Schüler keine wesentlichen Probleme, sich mit der Programmierumgebung zurechtzufinden, auch wenn sie vorher noch nie programmiert haben. Bedenken gab es u.a. zuerst, da die Bausteine in der Programmierumgebung nur in englischer Sprache zur Verfügung stehen. Mit Hilfe eines Beschreibungsblattes (vgl. Abb. 1) stellte dies für die Schüler aber keine Hürde dar.³ Die abwechselnde Arbeit sowohl an der Konstruktion und den einfachen Programmen erwies sich hinsichtlich der Konzentration der Schüler aufgrund der Verschiedenartigkeit der Tätigkeiten als vorteilhaft. Die von den Schülern geleisteten Programmierungen sind von geringer Komplexität, was auch zu erwarten war. Bei Rückfrage zur Bedeutung der einzelnen Programmstrukturen konnten die Schüler aber zumeist problemlos den Ablauf ihrer Programme erläutern. Insgesamt war festzustellen, dass, wie in o.g. Projekten, die Schüler und insbesondere auch die Mädchen viel Spaß an den Projekten hatten. Dieser Spaß am Kreieren und Programmieren kann genutzt werden, weiteres Interesse für die Informatik zu wecken.

Die Projekte beschränkten sich auf nur wenige Stunden in der Gruppe. Ein umfangreicher Einsatz und die Auswertung hinsichtlich der erworbenen Kompetenzen der Informatik sind basierend auf diesen ersten Erfahrungen geplant. Robotik mit PicoCrickets stellt sich nach diesen Erfahrungen für die Schüler der Grundschule als spannender, interessanter und kreativer Zugang zur Informatik dar.

Literaturverzeichnis

- [Bo05] Boles, D.: Spielerisches Erlernen der Programmierung mit dem Java-Hamster-Modell. In Lecture Notes in Informatics (LNI)-Proceedings 60, 2005; S. 243-252.
- [BD09] Borowski, C.; Diethelm, I.: Kinder auf dem Wege zur Informatik: Programmieren in der Grundschule. In (Peters, I.-R., Hrsg.): Informatische Bildung in Theorie und Praxis. Beiträge zur INFOS 2009. 13. GI-Fachtagung - Informatik und Schule., Berlin, LOG IN, 2009.
- [BBE09] Bruckman, A.; Biggers, M.; Ericson, B.; McKlin, T.; Dimond, J.; DiSalvo, B.; Hewner, M.; Ni, L.; Yardi, S.: Georgia computes!: improving the computing education pipeline. In ACM SIGCSE Bulletin 41(1), 2009; S. 86-90.

³ PicoBlocks ist nur in Englischer und Spanischer Sprache verfügbar. Eine Übersetzung ist grundsätzlich durch Editieren der entsprechenden Bilder möglich.

- [Fr09] Freudenberg, R.: Lernen mit Etoys. Proc. INFOS 2009, Berlin, 2009.
- [GI00] Gesellschaft_für_Informatik_e.V.: Empfehlungen für ein Gesamtkonzept zur Informatischen Bildung an allgemein bildenden Schulen, Bonn, 2000.
- [Gu10] Guzdial, M.: Dancing and singing humans, even more than robots. 2010. <http://computing.ed.wordpress.com/2010/12/31/> (10.01.2011).
- [Ha10] Hart, M. L.: Making contact with the forgotten k-12 influence: are you smarter than your 5th grader? Proc., ACM, 2010.
- [HH09] Herper, H.; Hinz, V.: Informatik im Primarbereich. Proc. INFOS 2009, Berlin, 2009.
- [Hi04] Hillen, A.: Computer in der Grundschule. Die Umsetzung des Projekts N21., Fachhochschule Braunschweig/Wolfenbüttel, 2004.
- [HL84] Hoppe, H. U.; Löthe, H.: Problemlösen und Programmieren mit Logo: Ausgew. Beispiele aus Mathematik u. Informatik. Teubner, 1984.
- [MW06] Maaß, S.; Wiesner, H.: Programmieren, Mathe und ein bisschen Hardware... Wen lockt dies Bild der Informatik? In Informatik-Spektrum 29(2), 2006; S. 125-132.
- [MKL10] Marcu, G.; Kaufman, S. J.; Lee, J. K.; Black, R. W.; Dourish, P.; Hayes, G. R.; Richardson, D. J.: Design and evaluation of a computer science and engineering course for middle school girls. Proc. SIGCSE 2010, Milwaukee, ACM, 2010.
- [MFS08] Medienpädagogischer_Forschungsverbund_Südwest: KIM-Studien. 1999-2008. <http://www.mpfs.de/index.php?id=10>
- [MIT11] MIT: Crickets (presentation). 2011. <http://llk.media.mit.edu/projects/cricket> (20.01.2011).
- [Pa82] Papert, S.: Mindstorms: Kinder, Computer und Neues Lernen. Birkhäuser Verlag, Basel, 1982.
- [Pa98] Papert, S.: Die vernetzte Familie. Kreuz Verlag, Stuttgart, 1998.
- [PH91] Papert, S.; Harel, I.: Situating Constructionism. In (Papert, S.; Harel, I., Hrsg.): Constructionism, Norwood, N.J., Ablex Publishing, 1991.
- [RNH04] Reichert, R.; Nievergelt, J.; Hartmann, W.: Programmieren mit Kara: ein spielerischer Zugang zur Informatik. Springer, 2004.
- [Re07] Resnick, M.: Sowing the Seeds for a More Creative Society. Proc. Learning & Leading with Technology, International Society for Technology in Education (ISTE), 2007.
- [RBM96] Resnick, M.; Bruckman, A.; Martin, F.: Pianos not stereos: Creating computational construction kits. In interactions 3(5), 1996; S. 40-50.
- [Ro08] Romeike, R.: Where's my Challenge? The Forgotten Part of Problem Solving in Computer Science Education. Proc. 3rd ISSEP Intern. Conf. on Informatics in Secondary Schools - Evolution and perspectives, Torun, Polen 2008, 2008.
- [RRB08] Rusk, N.; Resnick, M.; Berg, R.; Pezalla-Granlund, M.: New pathways into robotics: strategies for broadening participation. In Journal of Science Education and Technology 17(1), 2008; S. 59-69.
- [Sc01] Schwill, A.: Ab wann kann man mit Kindern Informatik machen? - Eine Studie über informatische Fähigkeiten von Kindern. Proc. INFOS2001-9. GI-Fachtagung Informatik und Schule, Paderborn, 2001.
- [We09] Weigend, M.: Algorithmik in der Grundschule. INFOS 2009. Berlin, 2009.
- [WB07] Wiesner, B.; Brinda, T.: Erfahrungen bei der Vermittlung algorithmischer Grundstrukturen im Informatikunterricht der Realschule mit einem Robotersystem. In Didaktik der Informatik in Theorie und Praxis, 2007; S. 113.
- [YB07] Yardi, S.; Bruckman, A.: What is computing?: Bridging the gap between teenagers' perceptions and graduate students' experiences. Proc., ACM, 2007.
- [Zi85] Ziegenbalg, J.: Programmieren lernen mit Logo. Hanser München, 1985.

Zur Didaktik der Algorithmik

Kerstin Strecker

Max-Planck Gymnasium
Theaterplatz 10
37073 Göttingen
kerstin.strecker@gmx.de

Abstract: In dieser Arbeit wird von der Tätigkeit des „Programmierens“ der Teil extrahiert, der sich auf die Algorithmik konzentriert und ein „roter Faden“ aufgezeigt, wie Schülerinnen und Schülern im Verlauf ihrer Schulzeit von der Grundschule bis zum Abitur das selbstständige Entwickeln von Algorithmen erlernen können. Dabei rückt neben dem klassischen systematisch-analytischen Zugang auch der experimentelle Zugang in den Blickpunkt der Betrachtungen, der nach Meinung der Autorin gerade bei jüngeren Schülerinnen und Schülern eine große Bedeutung im Lernverlauf hat.

1 Einleitung

Kinder in der dritten Klasse einer Grundschule lernen, große Zahlen schriftlich miteinander zu addieren. Dabei wird ihnen ein Algorithmus vorgesetzt, der in etwa folgende Struktur aufweist (siehe Abbildung 1):

Schriftliche Addition

Schreibe die zu addierenden Zahlen rechtsbündig untereinander	
von rechts nach links tue	
addiere die untereinanderstehenden Ziffern und einen dort eventuell stehenden Übertrag	
Summe > 10?	
ja	nein
Schreibe die Einerstelle der Summe in die Summenzeile unter die Ziffer	Schreibe das Ergebnis in die Summenzeile unter die Ziffer
Schreibe die Zehnerstelle als Übertrag zur Ziffer eine Stelle nach links	

Abbildung 1 Algorithmus zur schriftlichen Addition

Wir finden als Grundstruktur eine Schleife mit einer Alternative im Schleifenrumpf. Die Kinder wenden diesen Algorithmus erfolgreich an. Obwohl das Stellenwertsystem im Unterricht behandelt wurde, können die Kinder aber kaum erklären, *warum* dieser Algorithmus funktioniert. Unbestritten ist aber, dass das Anwenden eines Algorithmus dieser Struktur für Kinder der dritten Klasse angemessen ist¹.

Ebenfalls angemessen erscheint der Zentralabiturkommission in Niedersachsen das Fordern der Lösung folgender Aufgabe im Abitur:

„Implementieren Sie in Java bzw. Pascal / Delphi eine Operation `verschluessele`, die eine Verschlüsselung nach dem Verfahren von Vigenere durchführt. Die Eingabeparameter sind eine Zeichenkette `klartext` und eine Zeichenkette `schluesselwort`. Es soll eine Zeichenkette zurückgegeben werden, die den verschlüsselten Text enthält.“ [ZA09]

Und die Struktur der Lösung? Ein Algorithmus bestehend aus einer Schleife, mit innerer Alternative (wie in Abbildung 2 ersichtlich):

Vigenere - Verschlüsselung

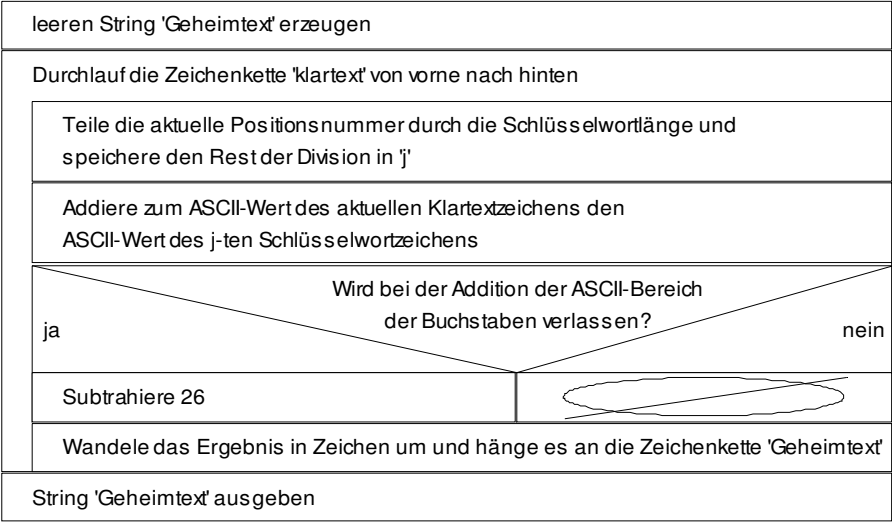


Abbildung 2 Vigenere-Verschlüsselung

¹Der Algorithmus der schriftlichen Multiplikation (Jg. 4) beinhaltet sogar doppelt geschachtelte Schleifen.

Von der Struktur her sind die beiden vorgestellten Algorithmen, der aus der Grundschule und der aus der Abituraufgabe, durchaus *vergleichbar*. Ihre Struktur liegt in ein und derselben **Komplexitätsklasse**.

Trotzdem würde natürlich niemand behaupten, dass sich zwischen der Grundschule und dem Abitur nichts verändert. Nur müssen im Mittelpunkt der Überlegungen nicht die Algorithmen, sondern der *Umgang* mit ihnen stehen.

Betrachten wir im Folgenden einige der hier aufgelisteten Kompetenzen, die man im Umgang mit Algorithmen identifizieren kann:

- ➔ Teilalgorithmen anwenden (ohne Rechner)
- ➔ Teilalgorithmen modifizieren
- ➔ Teilalgorithmen erklären
- ➔ Teilalgorithmen zusammensetzen / vernetzen
- ➔ Teilalgorithmen zielgerichtet entwickeln

Dann finden wir den Bereich „Teilalgorithmen anwenden“ in Klasse 3, aber auch im Abitur, wenn die Schülerinnen und Schüler in Teilaufgabe a) der oben gezeigten Abituraufgabe eine Vigenere-Verschlüsselung durchführen müssen; selbst im Studium finden sich in Klausuren Aufgaben, bestimmte Algorithmen auf Beispiele anzuwenden. Aufgaben im Bereich „Teilalgorithmen anwenden“ verschwinden also nicht völlig, nur ihre Gewichtung verändert sich mit dem Alter und der Erfahrung der Schülerinnen und Schüler. Je älter die Lernenden werden, desto mehr ist zusätzlich die Kompetenz des eigenständigen Algorithmenentwurfs gefordert.

Wie schaffen wir aber den Übergang vom stark Angeleitetsein zur Selbständigkeit bei gleichbleibender Komplexität der Algorithmen?

2 Modifikation vorgegebener Programme in realen Miniwelten

2.1 Jahrgang 3/4

Im Rahmen eines aktuellen Schulversuchs², verzeichnen wir Erfolge mit folgendem Konzept für die Klassenstufe 3/4 im Bereich Technik: Die Schülerinnen und Schüler bekommen eine reale Miniwelt präsentiert, eine kleine Stadt aus LEGO, mit Straßen, einer Eisenbahn, Häusern, Spielplatz, Bahnhof,... Zunächst bauen sich die Kinder auch ihre eigenen Häuser in die Stadt oder erweitern sie um Blumenbeete, Reitplatz, Feuerwehr,... da sich gezeigt hat, wie groß (gerade bei jungen Schülerinnen und Schülern) das Interesse hieran ist und auf diesem Weg ein Hereindenken in die Miniwelt

²Schulversuch „Naturwissenschaftliche und Technische Bildung am Übergang von der Primar- zur Sekundarstufe“

und eine Identifizierung mit dieser erfolgt. Jetzt wird WeDo [WD11] in Kombination mit „scratch“ [Sc11][SW11] verwendet, um einige Dinge in dieser Stadt zu automatisieren. Die Schranke geht hoch, wenn sich ein Auto nähert (Entfernungssensor, Motor,...), die Häuser werden beleuchtet, wenn jemand in kurzem Abstand vorbei geht, mit dem Kippsensor als Schalter wird die Luke für das Silo mit Streusalz geöffnet,..... Alle Programme in dieser Welt haben den Aufbau:

Wert von Sensor überschritten/ unterschritten → Motor an/aus für ... Sekunden, ...

Diese Programme sind fertig **vorgegeben**, die Automatisierung der LegoWelt funktioniert, nur eben nicht schön. Die Schranke öffnet sich nur zu 2/3, weil der Motor nicht lang genug läuft. In dem roten Haus wohnt eine Oma, die sehr langsam ist, weshalb die Beleuchtung länger eingeschaltet sein muss als anderswo oder der Entfernungssensor muss so eingestellt werden, dass nicht immer die Katze die Beleuchtung einschaltet, wenn sie aus ihrem Katzenhaus läuft,... Durch eine Modifikation der gegebenen Programme verbessern die Schülerinnen und Schüler jetzt ihre Welt. Sie verändern dabei aber nur folgende **Parameter**: Einmal die Schwellwerte in der Bedingung, die zum Sensor gehört und einmal die Zeitangaben oder Motorrichtungen der Aktionen, die ausgelöst werden.

Durch experimentelles Vorgehen in der Parametermodifikation, werden die Lernenden mit diesen immer strukturgleichen Programmen vertraut.

2.2 Jahrgang 5/6

In Klasse 5/6 wurden ebenfalls gute Erfahrungen damit gemacht, in realen Miniwelten WeDo einzusetzen. Hier mussten die Kinder aber anhand der Aufgabenstellung selbst entscheiden, welche Sensoren und Aktoren wo in der Welt platziert werden müssen. Die Programmstruktur wurde **fest vorgegeben**, wie in Abbildung 3 zu sehen.



Abbildung 3 vorgegebene Programmstruktur



Abbildung 4 Werkzeugkasten

Die Schülerinnen und Schüler mussten aus einem Werkzeugkasten (Abbildung 4) die richtigen Elemente entnehmen und aus den Aufgabenstellungen Sinn entnehmend erlesen, welche Bedingungen und Aktionen man an welcher Stelle einsetzt. Dabei waren alle Programme ebenfalls strukturgleich.

Durch die Eigenheiten der verwendeten Programmiersprache „scratch“, Syntaxfehler im Vorfeld zu vermeiden, erkennen die Kindern bei der Ergänzung der Bedingung beispielsweise, dass sie nur eine Wahl zwischen den „grünen“ Komponenten haben. Es muss nicht vollständig durchdrungen sein, dass im Kopf einer Alternative ein boolescher Ausdruck stehen muss; die Bedingung kann auch ergänzt werden, ohne dass die Systematik dahinter transparent gemacht wird.

Durch die vorgegebene Auswahl von Bausteinen, können Schülerinnen und Schüler das richtige Konstrukt nicht nur entnehmen, sondern die möglichen Konstrukte auch gegeneinander abwägen und aus *diesen* Überlegungen ein Konstrukt auswählen. Verdeutlicht werden soll das an einem Beispiel:

Ein Quiz, bei der der Kandidat eine Frage gestellt bekommt, die er ohne Hilfe beantworten muss, ist schwerer als ein Quiz, bei dem der Kandidat die Antwort auf eine Frage aus vier möglichen vorgegebenen Antworten heraussuchen muss, weil man die Möglichkeiten gegeneinander abwägen oder die falschen Antworten ausschließen kann.

Eine Ergänzung von Bedingungen und Anweisungen in Algorithmen einer vorgegebenen Struktur erfordert noch keine Durchdringung der Problemlösestrategie, wir sprechen eher von einer

Anpassung einer gegebenen Strategie an die konkrete Aufgabe.

3 Experimentelle Lösungszugänge in virtuellen Miniwelten

Folgende Ergebnisse zeigen ein Schulversuch³ und [St09]: In der Mittelstufe (Jahrgang 7/8/9) können den Kindern bereits Aufgaben gestellt werden, bei denen die Algorithmenstruktur *nicht* mehr vorgegeben ist, sondern von den Schülerinnen und Schülern aus vorgegebenen Algorithmenbausteinen zusammengesetzt werden muss. Ebenso müssen Anweisungen und Bedingungen ergänzt und eventuell ebenfalls zusammengesetzt werden. Richtig „offen“ sind die Strukturen dennoch nicht, weil die Schülerinnen und Schüler Alternativen und Schleifen nur ineinander verschachteln oder die bekannte Struktur um Alternativen und Schleifen nacheinander erweitern.

³Schulversuch „InTech – Informatik mit technischen Aspekten“ in Niedersachsen

Eine eigenständig entwickelte Schülerlösung in der Programmierumgebung „scratch“ zeigt Abbildung 5:



Abbildung 5 Programm eines Schülers einer 8. Klasse mit „scratch“

Wieder ein strukturgleicher Algorithmus, der hier zum ersten Mal von den Schülerinnen und Schülern selbst entwickelt wurde, ganz ohne vorgegebene Programmstruktur. Werkzeuge wie „scratch“ oder „Robot Karol“ [Rk11], „AutomatenKara“ [Ka11] und andere, die virtuelle kleine Miniwelten vorsehen, sind in diesen Jahrgängen weit verbreitet. **Diese Sprachen bieten alle Werkzeugkästen, integriert in ihre Programmierungsumgebungen.**

Lassen wir die Schülerinnen und Schüler selbst entscheiden welche Algorithmen sie implementieren wollen, dann werden fast in allen Jahrgangsstufen in „scratch“ gerne kleine „Spiele“ programmiert, wie „Pacman“, „Pong“ o.a. In der Mittelstufe können Aufgaben gelöst werden, bei denen die Algorithmenstruktur von den Schülerinnen und Schülern aus vorgegebenen Algorithmenbausteinen nach dem Baukastenprinzip zusammengesetzt werden muss. Durch Beobachtungen der Schülerinnen und Schüler müssen wir davon ausgehen, dass ein eigenständiges Finden von geeigneten Algorithmen zur Lösung einer Aufgabe mit den obigen Werkzeugen, zum Großteil durch **experimentelle** Herangehensweisen bestimmt ist.

Das Werkzeug „scratch“, wie auch andere, ermöglicht durch vielfältiges Kombinieren der einzelnen Bausteine eines Werkzeugkastens und dadurch, dass *keine Syntaxfehler* gemacht werden können und *immer ein ausführbares Programm* entsteht, einen experimentellen Zugang bei der Entwicklung eines Algorithmus. Die Kinder können Bausteine kombinieren, die entstehenden Algorithmen ausprobieren und zumindest Teile ihrer späteren Lösung durch Versuch und Irrtum entwickeln. Ihre korrekte Lösung entsteht u.a. durch ein Verwerfen von falschen Lösungen. Da die Bausteine und Konstrukte in ihrer Anzahl beschränkt sind, insbesondere Datenstrukturen keine Rolle spielen, ist dieser Zugang möglich.

Wenn erste Algorithmen so entwickelt werden, dann machen die Kinder Erfahrungen, die sie in späteren Algorithmen wieder verwenden. Dann systematisieren sie bei einer häufigen Suche verschiedenster Algorithmen ihre Erfahrungen und abstrahieren sie im optimalen Fall. Diese Kompetenz kann im Unterricht unterstützt werden. Indem man die Schülerinnen und Schüler in verschiedenen gleichartigen Lernumgebungen wie „AutomatenKara“, „Lego-Mindstorms“ [Le11], „scratch“, „Robot Karol“ u.a. Algorithmen entwickeln lässt, und anschließend im Unterrichtsgespräch systematisiert, dass alle Sprachen gleichartige Bausteine, wie „Schleife“, „Alternative“, ... enthalten, machen für die Lernenden auch Konstrukte wie Struktogramme oder UML-Aktivitätsdiagramme als Sprachen übergreifende Verbindung Sinn. In den Jahrgangsstufen 7, 8 und 9 steht also der experimentelle Zugang zur selbstständigen Algorithmensuche im Mittelpunkt. Genauer:

Verwendung von Lösungsstrategien, die durch ein neues Zusammensetzen bekannter Strategien entwickelt werden, mit teilweise experimentellem Probieren, bis der gewünschte Erfolg erreicht wird.

An dieser Stelle wird sich Widerspruch regen und der Wunsch laut werden, doch nicht noch zu fördern, dass die Lernenden „planlos“ irgendetwas zusammenklicken. Gewünscht ist doch der **analytische** Zugang. Das stimmt. Doch trial-and-error gibt es überall, zumindest ein Stück weit. Beobachtungen zeigen, dass der experimentelle Lösungszugang auch bei Informatik-Studenten nicht verschwunden ist und sich der Siegeszug der Sprachen, die diesen Ansatz unterstützen, vielleicht zum Teil auch mit diesem Argument erklären lässt.

Außerdem wird noch einmal ausdrücklich darauf verwiesen, dass die Sprachen, die einen experimentellen Ansatz unterstützen (eingeschränkte) *Werkzeugkästen* mit Algorithmenbausteinen enthalten müssen, die die Schülerinnen und Schüler syntaxfehlerfrei kombinieren können. **Jede beliebige Kombination muss dabei ein ausführbares** (jedoch nicht unbedingt semantisch korrektes) **Programm erzeugen**, damit die Schülerinnen und Schüler eine Reflexion aus dem Programm und der Beobachtung bei der Ausführung anschließen können. (Dies ist bei universellen Programmiersprachen nicht möglich. In Java z.B. ist eine Klassendeklaration im Schleifenrumpf nicht zulässig, nicht jede Kombination der Bausteine erzeugt also ein lauffähiges Programm) [St09].

4 Algorithmik mit universellen Sprachen

Die Algorithmen der Oberstufe sind nicht komplexer als in der Grundschule, wie bereits gezeigt wurde. Da sie aber auf beliebigen Datenstrukturen (Zeichenketten, Pixeln, Bäumen oder Datensätzen einer Datenbank beispielsweise) operieren, gelingt das eigenständige Entwickeln eines Algorithmus zur Lösung eines Problems nur dann, wenn die Schülerinnen und Schüler beim Entwickeln ihres Algorithmus genau durchdrungen haben, welche Semantik sich hinter welchem Konstrukt verbirgt. Durch die **Komplexität der Kombination von Möglichkeiten** wie z.B. in Java kann ein ausschließlicher experimenteller Zugang nicht mehr erfolgreich sein. Die Schülerinnen und Schüler müssen die Systematik der Algorithmik verstanden haben, die Struktur der Algorithmenbausteine muss durchdrungen sein, Probleme müssen analysiert werden können, um mittels eines analytischen Zugangs gezielt die Bausteine eines Algorithmus zusammensetzen zu können. Es reicht bei dem Schreiben einer Bedingung nicht nur, sich bewusst zu sein, dass hier ein boolescher Ausdruck folgen muss, man muss auch den Datenstrukturen entnehmen können, welche Methoden boolesche Ausdrücke liefern oder wie deren sonstige Rückgabewerte zu booleschen Ausdrücken verknüpft werden können. Durch das im obigen Kapitel skizzierte Verfahren der Systematisierung im Anschluss an eine Phase der konkreten Erfahrungen mit Algorithmen in den unterschiedlichsten Sprachen, kann der analytische Zugang gefördert werden [St09] und folgendes Ziel zumindest für die stärkeren Schülerinnen und Schüler auch erreicht werden:

Oberstufenschüler im Abitur entwickeln Lösungsstrategien m.E. auch gezielt.

5 Fazit

Zusammenfassend lassen sich also folgende Zugänge zur Algorithmik in den einzelnen Jahrgangsstufen herausfiltern:

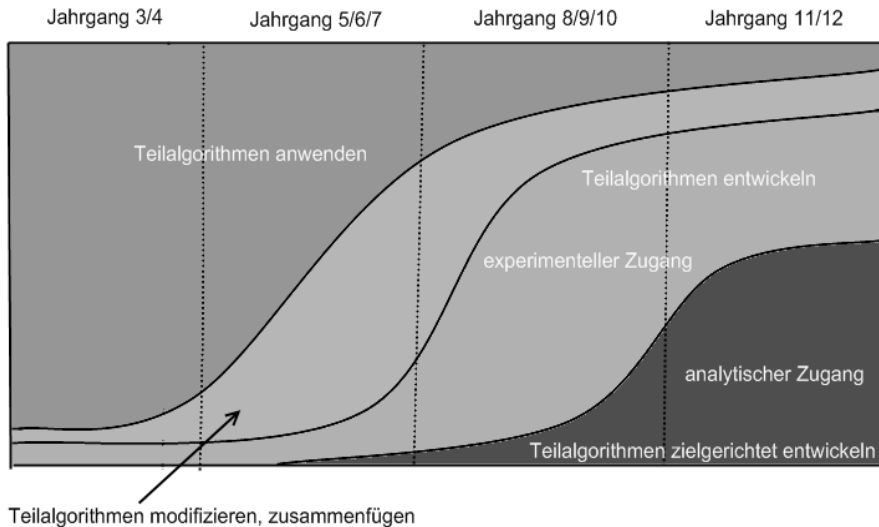


Abbildung 6 Algorithmenzugänge in den einzelnen Jahrgangsstufen

Die Kompetenzen:

- ➔ Teilalgorithmen anwenden
- ➔ Teilalgorithmen modifizieren, zusammenfügen
- ➔ Teilalgorithmen entwickeln mit dem experimentellen Zugang
- ➔ Teilalgorithmen (zielgerichtet) entwickeln mit dem klassischen systematisch-analytischen Zugang

wie in den einzelnen Kapiteln beschrieben, finden sich dabei unterschiedlich gewichtet in den verschiedenen Jahrgangsstufen. Dabei werden am Ende der Schulzeit immer noch **alle** Kompetenzen abgeprüft und verwendet. Und auch Schülerinnen und Schülern in Jahrgang 4 muss ermöglicht werden, bereits kleine eigene Algorithmen zu entwickeln.

Ein experimenteller Zugang zur Algorithmik verbietet nicht den analytischen Zugang, kann aber schwächeren Schülerinnen und Schülern helfen, überhaupt Algorithmen selbstständig entwickeln und nicht nur nachvollziehen zu können. Und dies ist für ein kreatives Fach Informatik (überlebens-)wichtig.

Literaturverzeichnis

- [ZA09] Niedersächsisches Kultusministerium: Aufgaben des Zentralabiturs Informatik 2009
- [Ka11] Raimond Reichert, Jürg Nievergelt, Werner Hartmann: „Programmieren mit Kara“, Springer-Verlag, 2005
- [Le11] URL: <http://education.lego.com/en-gb/preschool-and-school/secondary-11-18/11plus-lego-mindstorms-education/>, Zugriff: 9.1.2011
- [Rk11] URL: <http://www.schule.bayern.de/karol/download.htm>, Zugriff: 9.1.2011
- [Sc11] URL: <http://scratch.mit.edu/>, Zugriff: 9.1.2011
- [St09] Kerstin Strecker: „Informatik für Alle – wie viel Programmierung braucht der Mensch?“ Dissertation, Universität Göttingen, 2009, URL: <http://webdoc.sub.gwdg.de/diss/2009/strecker/>, Zugriff: 9.1.2011
- [SW11] URL: <http://info.scratch.mit.edu/WeDo>, Zugriff: 9.1.2011
- [WD11] URL: <http://www.lego-in-der-schule.de/lego-learning-blog-1/lego-education-wedo-robotics-ein-neues-konzept-fuer-robotik-in-der-grundschule>, Zugriff: 9.1.2011

Schülerinnen konstruieren informatische Bildung

Ludger Humbert

Bergische Universität Wuppertal
Fachbereich Mathematik und Informatik
Fachgebiet Didaktik der Informatik
humbert@uni-wuppertal.de

Abstract: Mit dem Ziel, bekannte Fallstricke für nicht gelingende informatische Bildung aus der über 40-jährigen Tradition des Informatikunterrichts in der Bundesrepublik Deutschland zu identifizieren, werden Hinweise zur fachdidaktisch qualifizierten Gestaltung entwickelt.

Dazu wird ein Feld unter den Pflug genommen, das offenbar ein ungeliebtes Kind der Informatik und erst recht der Informatikfachdidaktik darstellt: die Medienbildung. Die Konsequenzen unserer Überlegungen werden bis auf die konkrete Gestaltungsebene des Informatikunterrichts heruntergebrochen, um zu zeigen, wie informatische Bildung im Sinne einer Eigenkonstruktion aller Schülerinnen und Schüler gelingen kann.

1 Zieldimensionen informatischer Allgemeinbildung

1.1 Das Ziel – Durchsetzung des Grundrechts auf informatische Bildung

Die Zielrichtung der informatischen Allgemeinbildung besteht – trotz der hexadezimalen Organisationsstruktur des bundesdeutschen Bildungssystems¹ – in der Einlösung des Grundrechts auf Bildung (vgl. Artikel 14 der Charta der EU – Grundrechtskatalog, siehe auch [Ca01]) in unserer Welt: Schülerinnen und Schüler sind **ausnahmslos** so auf die unbekannte Zukunft vorzubereiten, dass sie diese verantwortlich für sich gestalten können (vgl.[GI08]).

In der Einlösung dieser Zielvorstellung sind verschiedene Konkretisierungen zu beobachten. Es beginnt mit der zunehmenden Verankerung eines Pflichtfachs Informatik für alle Schülerinnen aller Schulformen (und -stufen), geht über Wahlangebote im schulischen und/oder außerschulischen Bereich und endet bei Zertifikaten für Detailkenntnisse besonderer Ausprägungen von Informatikmitteln².

Die Differenzierung bei den Anforderungen geht so weit, dass einigen Ansätzen der Bezug zur Informatik kaum noch anzusehen ist. Beispiele dafür sind m. E. der ECDL (vgl.

¹ Chapeau an Steffen Friedrich für die Prägung »hexadezimale Struktur des Schulsystems«.

² Definition siehe [Hu06a, S. 75]

[DL10]) oder andere »Führerscheine«, die mit dem Ziel der ökonomischen Verwertung³ entwickelt und eingeführt wurden oder werden.

Mit der Dissertation von Siglinde Voß (vgl. [Vo06]) ist nachweisbar, dass bei Schulungen zur Nutzung konkreter Informatiksysteme durch Verzicht auf Bildschirmfotos und Details der Bedienung die Eigenkompetenz der Teilnehmenden entwickelt wird. Dies gilt (aber nur) unter der Voraussetzung, dass die hinter den Informatiksystemen stehenden informatischen Grundkonzepte verstanden werden. Nach der Vermittlung der grundlegenden Konzepte sahen sich die Teilnehmenden in der Rückschau besser in der Lage, ihre eigene weitere Arbeit mit Büropaketen auf einer informatischen Grundlage zu gestalten und nicht auf die nächste Schulung warten zu müssen, sobald Fragen auftauchten.

Zertifizierung von Detailbedienkenntnissen widersprechen dem Grundrecht auf informatische Bildung.

Damit muss die Zieldimension jeder allgemeinen Bildung auf das Schulfach Informatik bezogen werden: Es gilt, die Vermittlung der allgemeinbildenden, grundlegenden informatischen Konzepte so zu gestalten, dass es jedem Menschen möglich ist, bei konkreten Fragestellungen im Lebens- und Alltagszusammenhang handlungsfähig zu werden (und zu bleiben).

Beispiel: Informatische Literalität

Nehmen wir den Begriff »Literalität« wörtlich, so besteht die notwendige Kompetenz eines Gebildeten darin, konkrete fachliche Darstellungsformen auf Zeichen- und/oder Symbolebene zu verstehen. Dazu ist die mit einem Zeichen und/oder Symbol verbundene fachliche Sicht – also z. B. der Kontext und die Randbedingungen – zum Aufbau des Fachwissens geeignet zu nutzen. Dies wiederum ist Bestandteil der allgemeinen Bildung, d. h. es bedarf eines Lernorts in der Schule – typischerweise ein Schulfach, in dem die fachliche Sicht qualifiziert vertreten und fachdidaktisch angemessen für die Schülerinnen verfügbar gemacht wird.

Da viele Menschen mit Informatiksystemen (auch auf Zeichenebene) arbeiten, stellt sich die Frage, wie es um die informatische Literalität bestellt ist. In diesem Feld besteht dringender Forschungsbedarf: Bis heute finden sich in der Fachdidaktik Informatik keine aussagekräftigen und empirisch abgesicherten Beiträge, die sich forschungsleitend mit der Frage beschäftigen, wie beispielsweise die Vielfalt der Interpretation des Gleichheitszeichens in unterschiedlichen informatischen Kontexten für die Schülerinnen und Schüler erfahren wird und welche Konsequenzen die permanente Überladung des »=« mit immer wieder anderen, durchaus widersprüchlichen, kontextabhängigen Bedeutungen und damit Bedeutungsmustern hat.

Da dies kein Einzelfall ist, muss dringend herausgefunden (= untersucht) werden, welche informatisch geprägten literalen Welten Schülerinnen und Schüler mit der Semantik

³ Sowohl der Erwerb der notwendigen Materialien als auch die Abnahme der Prüfungen sind kostenpflichtig. Da die Materialien und Prüfungen regelmäßig an neue oder geänderte technische Bedingungen angepasst werden, geht diesen Gelddruckmaschinen der »Stoff« nicht aus.

der verschiedenen – nicht deckungsgleichen – anderen Zeichen im Informatikunterricht verbinden, wie sie diese trennen, vermischen und ob dies zu Lernwiderständen führt, die kontraproduktiv im Sinne unserer ausgewiesenen Zielperspektive wirken.

1.2 Medienbildung – Teil der informatischen Bildung?

Der Medienbegriff ist abhängig von den jeweiligen Bezugswissenschaften definiert. Die folgende Begriffsbestimmung wird im weiteren Text zugrunde gelegt.

Medien fungieren als Vermittler – Medien speichern, verarbeiten, übertragen Daten und/oder Wissen.⁴

Der damit angebotene Medienbegriff verdeutlicht, dass die Mittel, mit denen technisch elaborierte Medien realisiert werden, Informatikmittel sind, da diese dem zweiten Teil der Begriffsbestimmung deutlich entsprechen. Der Zusammenhang zwischen dem Anspruch der Medienbildung und der informatischen Allgemeinbildung erschließt sich – so wir der obigen Begriffsbildung folgen – theoriegeleitet über die in der Informatik vorgenommene Modellierung, da nur diese den Zugang zur Gestaltung und damit den geplanten und beabsichtigten Anwendungsfällen von Informatikmitteln erlaubt.

Jeder Ansatz einer Medienbildung, der primär über die technische – nicht aber die informatische – (Aus-)Prägung der zur Anwendung gebrachten Elemente gestaltet wird, bleibt »in der Oberfläche stecken«. Soll Medienbildung auch eine gestaltende Kraft entfalten können muss diese Sicht zugunsten der informatischen Modellierung aufgebrochen werden.

In dem Feld der Medienbildung – verstanden als Erweiterung der informatischen Allgemeinbildung – besteht Handlungsbedarf. Da hilft kein Wegschauen: Die Anreicherung unserer Umwelt mit immer mehr – häufig unsichtbaren – Informatiksystemen führt dazu, dass zunehmend deutlich wird: Das Herz der Medien besteht in der informatischen Modellierung, die dem jeweiligen System zugrundeliegt. Wir Informatiklehrende können nicht aus unserer Verantwortung entlassen werden, wenn es um Medien geht – wir müssen deutlich machen, dass wir die hinter den Medien stehende informatische Modellierung durchschauen, durchschaubar machen können und konstruktiv Wege angeben, wie aus Nutzenden Gestaltende werden können.

Diese Sicht betrifft nicht nur die seit 40 Jahren so genannten »Neuen« Medien, sondern auch die »Alten« Medien, werden doch gerade dort in der Produktionskette Informatikmittel eingesetzt – insbesondere für bestimmte Elemente, die für die Medienproduktion unerlässlich sind, werden Informatiksysteme eingesetzt. Die Wege zu einer an der Idee **echter Gestaltung** orientierten Medienbildung führen allesamt über eine qualifizierende informatische Bildung. Die dazu notwendige informatische Modellierung (inkl. Implementierung) kann nicht durch Bedienkenntnisse ersetzt werden. Daraus ist abzuleiten, dass jede Medienbildung auf informatischer Allgemeinbildung aufgebaut werden muss.

⁴ In diesem Beitrag werden die Begriffe Information, Wissen und Daten in der Form verwendet, wie sie in [Hu06a, S. 10ff] definiert sind.

2 Der Weg zur informatischen Vernunft

Alle Schülerinnen entwickeln ihre Informatikkompetenzen, indem sie informatisches Fachwissen (zu den Begriffen: Information, **Wissen** und Daten vgl. Fußnote 4) erarbeiten und sich damit Wissensgebiete der Informatik erschließen. In der Bearbeitung realweltlicher Phänomene und Situationen mit informatischem Fachwissen entstehen bei Schülerinnen zunächst Wissensinseln, die durch eine fachliche Klammer verbunden und vernetzt werden müssen. Bei der Umsetzung dieser Forderungen stoßen wir auf Probleme, die der dringenden Bearbeitung seitens der Informatikfachdidaktik bedürfen.

2.1 ... gepflastert mit Fehlvorstellungen

Wir müssen feststellen, dass in die Konstruktionen der Informatikwelt der Schülerinnen dramatische Fehlvorstellungen eingebaut werden: Beispiele: die mangelnde Unterscheidung zwischen verschiedenen Abstraktionsebenen wie am Beispiel Objekt – Klasse nachgewiesen wurde (vgl. [Hu06a, S. 160ff]) oder die »if-Schleife« (vgl. [Hu06b]). In diesen und ähnlichen Fällen hat die Didaktik der Informatik Ansätze zu entwickeln, zu diskutieren und umzusetzen, die es Informatiklehrkräften ermöglichen, den Weg **aller** Schülerinnen zur informatischen Allgemeinbildung erfolgreich zu gestalten.

Die Voraussetzungen für die Einlösung dieser Forderung bestehen in dem Ausweis der informatischen Grundlagen und der Entwicklung einer fachdidaktisch gestaltungsfähigen Umsetzung der Anforderungen auf einer soliden Grundlage: Nachweise zur Eignung müssen sowohl auf einer fachdidaktisch und lerntheoretisch abgesicherten wie auch auf einer empirischen Basis erfolgen.

2.2 Verschränkung der informatischen Bildung mit der Medienbildung

Es soll der Versuch unternommen werden, die informatische Allgemeinbildung im Sinne der Medienbildung wirksam werden zu lassen. Dazu schlagen wir den Aufbau einer Referenzliste vor, die verdeutlicht, wie Inhalts- und Prozessbereiche der informatischen Allgemeinbildung als grundlegende Elemente zur Medienbildung beitragen. Inhalts- und Prozesskompetenzen der informatischen Bildung sind in [GI08] ausgewiesen (vgl. Abbildung 1) und bis auf die Ebene der Jahrgänge (jeweils Cluster für die Jahrgangsstufen 5 bis 7 und 8 bis 10) differenziert dargestellt.

Für die Medienbildung wurde mit [Sc09] der Bericht einer Expertenkommission für das Bundesministerium für Bildung und Forschung vorgelegt, der eine Liste von **Themen- und Aufgabenfeldern** präsentiert, die in der Abbildung 2 dokumentiert sind. Die in beiden Listen auftretenden Punkte haben wir miteinander verbunden. Die Richtung der Auszeichnung der Verbindung drückt aus, welches Element der informatischen Allgemeinbildung nach unserer Einschätzung die Voraussetzung für das zugeordnete Element der Medienbildung liefert. Durch die Abbildungen 1 und 2 und ihre von uns explizierten Verbindun-



Abbildung 1: **Informatik** – Inhalts- und Prozesskompetenzbereiche lt. [GI08, S. 11]

gen wird ohne jeden Zweifel deutlich, dass eine enge Kopplung zwischen der informati-
schen Allgemeinbildung und der Medienbildung besteht. Dem Expertenbericht kann keine
Zuordnung von Detailkompetenzen auf Jahrgangsstufenebene, wie in [GI08] entnommen
werden. Wir versuchen, Ideen aus [Sc09] zu extrahieren, die verdeutlichen, wie die Infor-
matikfachdidaktik zentrale Elemente der Medienbildung vorbereiten und umsetzen kann.

Die Expertengruppe konstatiert »Einige Bundesländer haben verpflichtend einen Informa-
tikunterricht eingeführt, der jedoch im Hinblick auf ein umfassend zu betrachtendes Me-
dienhandeln nicht weit genug greift« [Sc09, S. 9]. Die aus dieser Feststellung zu schließende
Notwendigkeit einer erweiterten informatischen Allgemeinbildung wird allerdings
nicht expliziert. Der vorgelegten Darstellung zu den Punkten 3 und 4 (vgl. Abbildung 2) in
[Sc09, S. 5ff] können wir entnehmen, dass Elemente für notwendig erachtet werden, die
zentral dem Bereich der Persönlichkeitsbildung zugeordnet werden.

Betrachten wir exemplarisch eine der sechs von der Expertengruppe ausgewiesenen Kom-
petenzen zum Themen- und Aufgabenfeld »Identitätssuche und Orientierung« (vgl. Ab-
bildung 2): »Problemlösung durch experimentelles und spielerisches Vorgehen mit dem
Erwerb von systematischen Zugängen verbinden« (vgl. [Sc09, S. 6]). Wie soll diese Kom-
petenz, die im Rahmen der INFOS 1997 in Duisburg mit [Kr97] in die Diskussion der In-

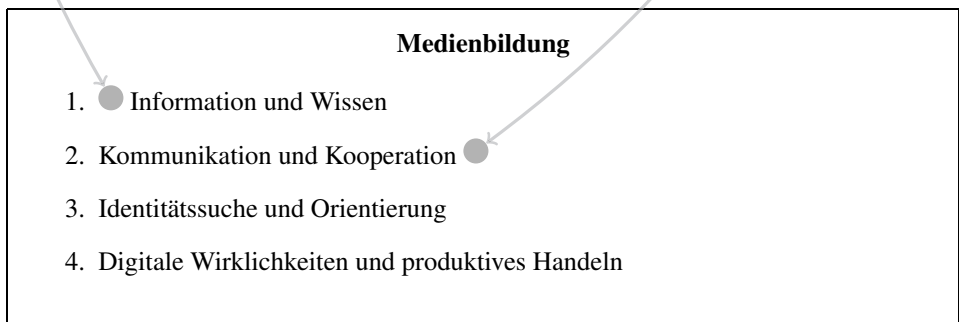


Abbildung 2: **Medienbildung** – Themen- und Aufgabenfelder lt. [Sc09, S. 3]

formatikfachdidaktik eingebracht wurde, ohne einen grundlegenden Bezug zur Informatik erzielt werden? Im Zusammenhang mit Kompetenzen aus dem Prozessbereich »Modellieren und Implementieren« (vgl. Abbildung 1) findet der Punkt seine Entsprechung in der informatischen Bildung. Die weitere Untersuchung der Vorschläge aus dem Expertenbericht führen uns zu der Feststellung:

Medienbildung ohne informatische Bildung ist unmöglich – der Umkehrschluss ist allerdings unzulässig.

3 Eine Variante zur Umsetzung

Um die in diesem Beitrag geforderte und begründete Verschränkung der informatischen Allgemeinbildung mit der Medienbildung einzulösen, bedarf es einiger konzeptioneller Vorüberlegungen: Es wird ein geeigneter Lernort im Rahmen der allgemeinen Bildung benötigt, die Ressourcenfrage/n sind zu klären und bei Bedarf anzupassen. Mit ausreichenden Ressourcen kann in einer Pilotphase der Aufbau einer Struktur für den kompletten Jahrgang an einer Schule entwickelt und durchgeführt werden. Parallel zu diesem Prozess sind die Ergebnisse einer begleitenden Evaluation zu unterziehen, um für nachfolgenden Gruppen zeitnah Konsequenzen ziehen zu können.

3.1 Kooperation zwischen Universität und Schule

Im Rahmen einer Kooperation zwischen dem Fachgebiet Didaktik der Informatik der Bergischen Universität Wuppertal und dem Carl-Duisberg-Gymnasium in Wuppertal wird zur Zeit der Versuch zur Umsetzung durchgeführt. Die Schule hat – aus eigenem Antrieb – den Weg zur Universität gefunden und um Unterstützung nachgefragt. Die Ressourcen der Schule können von Lehramtsstudierenden (neben den zu erwerbenden Leistungspunkten) zur Materialentwicklung und zur Erprobung der entwickelten Elemente im Unterricht und in unterrichtsfreien Zeiten sowie zur Unterstützung der Evaluation eingesetzt werden. Nun gilt es, eine beiden Seiten genügende Gestaltung konkreter Elemente vorzunehmen.

3.2 Elemente der Gestaltung

Randbedingungen für die Kooperation aus Sicht der Schule: Die Kooperationsschule unternimmt den Versuch, in den Jahrgängen 5 und 7 das für Berufskollegs entwickelte Modell des »Computerführerscheins« (vgl. [Ze07]) umzusetzen.

Auf universitärer Seite besteht das Interesse, die mit [LM07] dokumentierten Ideen für eine Umsetzung so zu gestalten, dass Informatikelemente ausgewiesen und unterrichtlich

konkretisiert werden (Hinweis am Ende der Präsentation [HM11]). Darüber hinaus werden die in dem Projekt »Mobiles Programmieren« gewonnenen Erkenntnisse zum Einsatz von Mobiltelefonen im Informatikunterricht der Sekundarstufe II (vgl. [Lö10]) auf ihre Übertragbarkeit für die Sekundarstufe I untersucht.

Daraus erwachsende Gestaltungsanforderungen: Um mit Projektpartnern – jenseits prototypischer Szenarien – gemeinsam Ansätze weiterzuentwickeln, wird eine intensivere lokale Kopplung zwischen Universitäten und Schulen realisiert. Diese Kooperation wird mit regelmäßigen Treffen der Beteiligten zur inkrementellen Weiterentwicklung der konzeptionellen Grundlagen der Informatikfachdidaktikgruppe verbunden. Im Spiegel der Umsetzung der Forschungsergebnisse – verknüpft mit der Reflexion der Lehrkräfte – wird die informatische Allgemeinbildung für alle Schülerinnen und Schüler des Kooperationspartners ausgestaltet.

Alle Materialien (sowohl für die Lehrkräfte als auch für die Schülerinnen und Schüler) werden so gestaltet, dass eine Umsetzung mit Mobiltelefonen möglich ist. Im Sinne der Untersuchungen zu Genderaspekten werden – vor allem in den Jahrgangsstufen der Unter- und Mittelstufe – Mobiltelefone eingesetzt. Die dazu notwendige Materialentwicklung wird Elemente der Bildungsstandards Informatik umfassen.

Genderaspekte werden in den Blick genommen, so dass spezifischen Forschungsinteressen

Jahrgang/Halbjahr	Betriebssysteme	Vernetzte Systeme	Datenschutz und Datensicherheit	Dokumente strukturieren	Hardware	Daten sammeln und auswerten	Summe
5.1	✓	✓	✓				3
5.2	✓			✓	✓		3
6.1		✓		✓	✓		3
6.2			✓		✓		2
7.1		✓	✓			✓	3
7.2				✓		✓	2
8.1	✓			✓			2
8.2	✓					✓	2
9.1		✗	✗				2
9.2				✗			1
10.1	✗				✗		2
10.2						✗	1
Summe	5	4	4	5	4	4	

Tabelle 1: Erster Vorschlag für eine Verteilung von Modulen auf Schulhalbjahre

Rechnung getragen wird.

3.3 Modulplan

Es werden Module ausgewiesen, die geeignet sind, um Schülerinnen und Schülern die informatische Allgemeinbildung zu ermöglichen. Ein Nebenziel besteht darin, dass alle Elemente **auch mit Mobiltelefonen** umsetzbar sein sollen. Jedes Modul soll soweit in sich abgeschlossen sein, dass es zu prüfbaren Kompetenzen führt. Im Unterricht des Differenzierungsbereichs kann die Erprobung verschiedener Module erfolgen. Die grundlegende Struktur der Module umfasst folgende Elemente:

1. Struktur des Gegenstands aus informatischer Sicht inkl. informatischer Modellierung
2. Implementierung/Programmierung
3. Nutzung im Zusammenhang mit Informatiksystemen

Module – Kurzbeschreibung – Umfang

Um die Kompetenzen zu erreichen, wird eine Umsetzung in Form von Teilmodulen vorgeschlagen. Die Modularisierung sollte dem Gedanken Rechnung tragen, dass die gewählten Module eine gewisse Unabhängigkeit voneinander besitzen. Dies ist nicht einfach zu bewerkstelligen, da einige der Module Voraussetzung für andere Module darstellen. In der Arbeit mit den Studierenden wurden gegenüber der in der Tabelle 1 dargestellten Modulverteilung sinnvolle Zusammenfassungen vorgenommen. Das Zwischenergebnis wird in Abbildung 3 als Modulübersicht dargestellt. Die Erarbeitung der Teilmodule aus dem Mo-

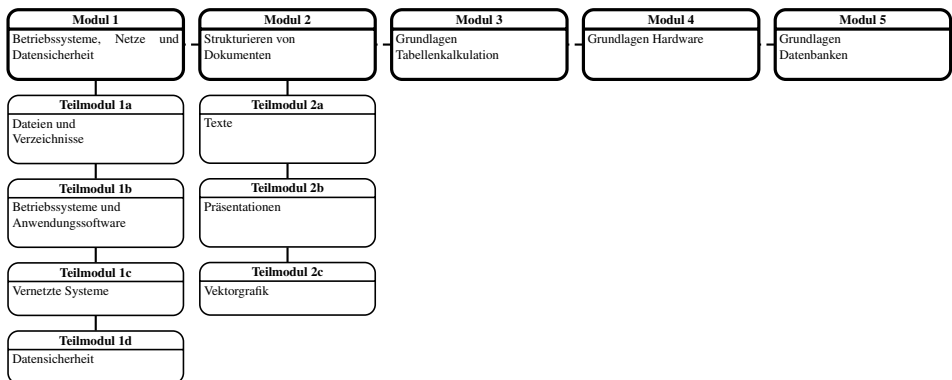


Abbildung 3: **Modulübersicht** – erste inhaltliche Detaillierung – Daniel Spittank (Lehramtsstudierender und Mitarbeiter im Kooperationsprojekt)

dul 1 **Betriebssysteme, Netze und Datensicherheit** stellt eine notwendige Voraussetzung zur praktischen Arbeit mit Informatiksystemen dar.

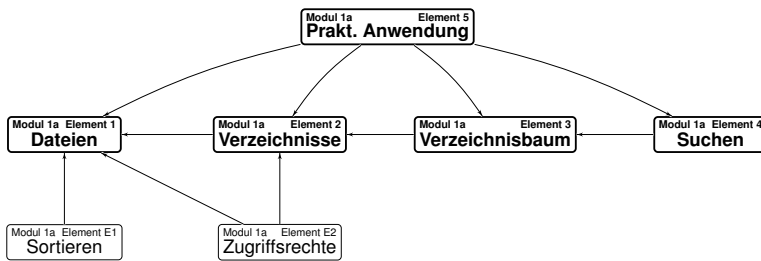


Abbildung 4: **Abhängigkeitsgraph – Dateien und Verzeichnisse** – Daniel Spittank

Die quantitative Ausprägung der einzelnen Module muss an den Kompetenzen orientiert werden, die Ziel des Moduls sind. Selbstverständlich ist es unabdingbar, dass die Elemente einzelner Module im konkreten Anwendungszusammenhang wieder thematisiert werden. Die jeweilige Modulabschlussprüfung kann stattfinden, sobald alle Teile des Moduls bearbeitet sind. Dies wird in der Tabelle 1 durch das Symbol $\triangleleft$ verdeutlicht.

Die Verteilung stellt einen ersten Versuch dar, die Inhalte so zu verzahnen, dass »Lernen auf Vorrat« vermieden wird. Die Darstellung macht darüber hinaus deutlich, dass die Module so eng gekoppelt sind, dass sie kaum unabhängig voneinander unterrichtlich verantwortlich realisiert werden können. Möglicherweise kann (ab Jahrgang 7) ein Modulaustausch realistisch sein.

Werden Module ausgewiesen, so ist es unabdingbar, die Werkzeuge zu nennen, mit denen die Module auch praktisch umgesetzt werden können. Im Unterschied zu anderen Vorschlägen ist es durch unseren Vorschlag (durch Nutzung der Programmiersprache Python) möglich, in jedem der Module auch die fachliche Modellierung bis hin zu ablauffähigen Programmen vorzunehmen.

Programmierarbeiten sollen dabei helfen, die Elemente vertiefend zu thematisieren, die durch einfache – typischerweise im Zusammenhang mit einer objektorientierten Modellierung gewonnene – Strukturen altersgerecht umgesetzt werden können.

Ich bedanke mich bei den (auch ehemaligen) Informatiklehramtsstudierenden der Bergischen Universität Wuppertal, bei Wolfgang Coy, der durch seine Mitarbeit bei der Enquete-Kommission »Internet« des Bundestages mir eine »Steilvorlage« zur Beschäftigung mit dem Spannungsfeld Informatische Bildung und Medienbildung gegeben hat, sowie bei den anonymen Begutachtenden für die Kommentare zu früheren Fassungen dieses Beitrages.

Literatur

- [Ca01] Johannes Caspar. Das Grundrecht auf Bildung in der EU. *European Journal for Education Law and Policy*, 5:21–29, 2001.
- [DL10] DLGI. Europäischer Computer Führerschein – Syllabus ECDL 5.0, Dezember 2010. DLGI – Dienstleistungsgesellschaft für Informatik.
- [GI08] GI. Grundsätze und Standards für die Informatik in der Schule – Bildungsstandards Informatik für die Sekundarstufe I, April 2008.
- [HM11] Ludger Humbert und Benedikt Mitscher. Freie Universität Berlin – 10. Tagung der GI-Fachgruppe »Informatik-Bildung in Berlin und Brandenburg« Workshop »Informatik und Medienkompetenz« – Präsentation, März 2011.
- [Hu06a] Ludger Humbert. *Didaktik der Informatik – mit praxiserprobtem Unterrichtsmaterial*. Leitfäden der Informatik. B.G. Teubner Verlag, Wiesbaden, 2., überarbeitete und erweiterte Aufl., August 2006.
- [Hu06b] Ludger Humbert. Informatische Bildung – Fehlvorstellungen und Standards. In Marco Thomas, Hrsg., *MWS – Münsteraner Workshop zur Schulinformatik 2006*, Seiten 37–46, Münster, Mai 2006. Westfälische Wilhelms-Universität.
- [Kr97] Sybille Krämer. Werkzeug – Denkzeug – Spielzeug. Zehn Thesen über unseren Umgang mit Computern. In Heinz Ulrich Hoppe und Wolfram Luther, Hrsg., *Informatik und Lernen in der Informationsgesellschaft*, Informatik aktuell, Seiten 7–13, Berlin, Heidelberg, September 1997. Springer.
- [LM07] LfM. jam! – Jugendliche als Medienforscher, 2007. LfM – Landesanstalt für Medien Nordrhein-Westfalen.
- [Lö10] Susanne Löffler, Dorothee Müller, Janin Panske, Matthias Heming und Ludger Humbert. Artefakte und Genderladung – Konsequenzen für den Informatikunterricht? *magazIn – halbjährliches Magazin der Gleichstellungsbeauftragten der Bergischen Universität Wuppertal*, 4(Wintersemester 2010/11):29–34, Oktober 2010.
- [Sc09] Heidi Schelhowe, Silke Grafe, Bardo Herzig, Jochen Koubek, Horst Niesyto, Antje vom Berg, Wolfgang Coy, Heinz Hagel, Joachim Hasebrook, Kurt Kiesel, Gabi Reinmann und Markus Schäfer. Kompetenzen in einer digital geprägten Kultur. Medienbildung für die Persönlichkeitsentwicklung, für die gesellschaftliche Teilhabe und für die Entwicklung von Ausbildungs- und Erwerbsfähigkeit. Bericht der Expertenkommission des BMBF zur Medienbildung, März 2009. BMBF – Bundesministerium für Bildung und Forschung.
- [Vo06] Siglinde Voß. *Modellierung von Standardsoftwaresystemen aus didaktischer Sicht*. Dissertation, Technische Universität – Institut für Informatik, München, Juni 2006.
- [Ze07] Detlef Zech. Sekundarstufe II – Berufskolleg; Zertifizierung von EDV-Kenntnissen im Berufskolleg – Entwurf, April 2007.

Maschinelle Erfassung von Problemlösestrategien bei algorithmischen Problemstellungen am Beispiel des Sortierens

Christian Wach

Technische Universität Darmstadt
Didaktik der Informatik
Hochschulstr. 10
64289 Darmstadt
wach@cs.tu-darmstadt.de

Abstract: Die Untersuchung von kognitiven Problemlöseprozessen mit Protokollanalysen ist sehr aufwändig. Mit dem Einsatz von maschinellen Lernverfahren verbindet sich die Hoffnung, aus leicht beobachtbaren Daten auf kognitive Prozesse schließen zu können.

Für Sortierprobleme wurden über 4000 unklassifizierte Nutzerinteraktionen und 490 klassifizierte Nutzerinteraktionen aufgezeichnet und ausgewertet. Verschiedene Klassifikationsverfahren wurden mit den klassifizierten Datensätzen evaluiert. Varianten der linearen Diskriminanzanalyse erreichten nicht nur eine geringe Fehlerrate bei der Klassifikation, sondern können auch zur Dimensionsreduktion der Datensätze genutzt werden.

1 Einleitung

Der Begriff des Problemlösens bzw. der des problemlösenden Denkens subsumiert in der kognitiven Psychologie die kognitiven Prozesse eines Lebewesens zur Lösung von Problemen. Folgende Definition von Dunker [Du35] hält sich in Varianten bis heute (vgl. [Fu03]):

„Ein *Problem* entsteht z. B. dann, wenn ein Lebewesen ein *Ziel* hat und *nicht weiß, wie es dieses Ziel erreichen soll*. Wo immer der gegebene *Zustand* sich nicht durch bloßes Handeln (Ausführen selbstverständlicher Operationen) in den erstrebten Zustand überführen lässt, wird das Denken auf den Plan gerufen. Ihm liegt es ob, ein vermittelndes Handeln allererst zu konzipieren“ [Du35] zit. nach [Fu03].

Das Problemraummodell (vgl. [NS72]) formalisiert den Problemlöseprozess. Mögliche Handlungen werden durch Operatoren abgebildet. Ein Operator überführt einen vorliegenden Problemzustand in einen neuen Problemzustand. Eine Lösung entspricht einer Folge von Operatoren, die den Anfangszustand durch Hintereinanderausführung in den Zielzustand überführen.

Nicht alle Probleme sind einander gleich. Daher wurden im Laufe der Zeit verschiedene Klassifikationen und Taxonomien vorgeschlagen (s. [Fu03, Kap. 1.4]). Schüler lösen im Unterricht oft *einfache* Probleme. Bei einfachen Problemen sind die Operatoren bekannt, die Anzahl der jeweils anwendbaren Operatoren ist gering und Anfangs- und Zielzustand sind genau definiert. Charakteristisch für *komplexe* Probleme ist nicht nur eine größere *Komplexität* (im Sinne eines größeren Problemraumes), sondern auch Unklarheit im Bezug auf Problemzustände und Operatoren (vgl. [Fu03]).

In der Informatik erhält der Begriff des Problemlösens eine zusätzliche Dimension. Mit der fundamentalen Idee der Algorithmisierung „verbindet sich die Zielvorstellung (Zielkriterium), alle Probleme ließen sich durch maschinell nachvollziehbare Verfahren, deren Korrektheit jederzeit gesichert ist, effizient lösen“ [SS04, Kap. 3.3.2]. Ein Algorithmus ist eine (formalisierte) Handlungsvorschrift zur Lösung eines Problems. Die Erarbeitung bzw. Herleitung einer solchen Handlungsvorschrift zur Lösung eines Problems ist im Regelfall ein zusätzliches kognitives Problem für den Schüler. Beispielaufgaben für unterschiedliche Jahrgangsstufen finden sich in den Bildungsstandards [Pu07].

Die Untersuchung von kognitiven Prozessen ist recht aufwändig. Mit qualitativen Sprachprotokollanalysen und der Think-Aloud-Methode können Rückschlüsse auf die kognitiven Prozesse gezogen werden (vgl. u.a. [RJJ10]). Mit der Anwendung von maschinellen Lernverfahren ist die Hoffnung verbunden, Rückschlüsse auf kognitive Prozesse aus relativ einfach beobachtbaren Daten, wie den Interaktionen mit Computersystemen, zu ziehen. Kiesmüller et. al. [Ki10a, Ki10b] untersuchten die Interaktionen von Schülern mit der Programmierlernumgebung Kara. Mit einem Hidden Markov Modell konnten die Interaktionen vier definierten allgemeinen Problemlösestrategien zugeordnet werden. McQuiggan et. al. [MML08] nutzten Verfahren des maschinellen Lernens, um aus erhobenen Daten Rückschlüsse auf die Selbstwirksamkeitserwartung einer Person zu ziehen.

1.1 Las Vegas Cardsort

Las Vegas Cardsort [WG09] wurde an der Technischen Universität Darmstadt von Studierenden im Rahmen eines Praktikums implementiert. Ziel der Entwicklung war ein System, das Schülern unterschiedlicher Jahrgangsstufen einen intuitiven und handlungsorientierten Zugang in das Themenfeld Sortieralgorithmen bietet.

Für die vorliegende Arbeit wurde Las Vegas Cardsort in der *Nur Tausch* Variante verwendet. Die Aufgabe für die Schüler bestand darin, 10 Karten mit Zahlen von 1 bis 99 aufsteigend zu sortieren. Für jeden Versuch wurden die Kartenwerte und die Vorsortierung der Karten zufällig bestimmt. Die Karten liegen verdeckt auf dem Spielfeld und müssen aufgedeckt werden. Jedoch kann nur eine Karte gleichzeitig sichtbar sein. Die Karten können nur paarweise vertauscht werden, die in Las Vegas Cardsort möglichen Einfügeoperationen wurden deaktiviert. Bei der Tauschoperation sind bis zu zwei Karten aufgedeckt.

Las Vegas Cardsort protokolliert für jeden Versuch den Anfangszustand und jede Nutzeraktion (Tausch, Ansicht) mit Zeitstempel in ein XML Protokoll.

1.2 Zielsetzung

Für die o.g. Konfiguration von Las Vegas Cardsort konnten bisher über 4000 erfolgreiche Sortierversuche erfasst werden. Auf Basis dieser Daten sollen unter Verwendung von Verfahren des maschinellen Lernens die folgenden Forschungsfragen beantwortet werden:

1. Können Handlungsfolgen von Schülern maschinell einer konkreten Lösungsstrategie zugeordnet werden?
2. Welche Lösungsstrategien können identifiziert werden?
3. Wie entwickelt sich die Lösungsstrategie eines Schülers, wenn dieser mehrfach Sortierungsprobleme löst?

2 Vorverarbeitung der Protokolldaten

Bevor die Protokolldaten analysiert und in maschinellen Lernalgorithmen verarbeitet werden können, müssen diese zunächst in eine geeignete Form gebracht werden. Bei der Vorverarbeitung ist insbesondere darauf zu achten, dass die Datensätze trotz unterschiedlichem Anfangszustand, bedingt durch die unterschiedliche Vorsortierung, vergleichbar bleiben. Um die Vergleichbarkeit zu gewährleisten, wurden die Karten nach ihrer jeweiligen Position im sortierten Zielzustand indiziert. Folglich ist die 0-te Karte die Karte mit dem kleinsten Wert und die 9-te Karte die Karte mit dem größten Wert. Auf Basis dieser Indizierung wurden die beobachteten Variablen in Tabelle 1 ermittelt.

<i>Variable</i>	<i>Beschreibung</i>
$t_i, i \in [0, 9]$	Zeitpunkt zu dem die i -te Karte zuletzt bewegt wurde, dividiert durch d .
$mc_i, i \in [0, 9]$	Anzahl der Vertauschungen der i -ten Karte
$vc_i, i \in [0, 9]$	Anzahl der Aufdeckungen der i -ten Karte
$m_i, i \in [0, 9]$	Anzahl der Vertauschungen bis zum Zeitpunkt t_i
$v_i, i \in [0, 9]$	Anzahl der Aufdeckungen bis zum Zeitpunkt t_i
p	längste Pause zwischen zwei Aktionen
l	Es seien i, j die Indizes zweier zu vertauschender Karten. l entspricht der Summe der Abstände $ i - j $ für alle Tauschoperationen
l_s	Summe der quadrierten Abstände $ i - j ^2$ für alle Tauschoperationen
s_m	Gesamtanzahl der Vertauschungen
s_v	Gesamtanzahl der Aufdeckungen
d	Benötigte Zeit zum Sortieren der Folge

Tabelle 1: Beobachtete Variablen nach der Vorverarbeitung

Der erste Schritt der Vorverarbeitung erfolgte in Java. Hierzu wurden anhand der Protokolldateien die Handlungsfolgen nachgespielt und hierbei die Variablen aufgezeichnet. So können die Variablen nicht nur im Nachhinein aus den Logdateien berechnet, sondern auch in Las Vegas Cardsort direkt bestimmt werden. Die Variablen werden als CSV Datei exportiert und können so in anderen Umgebungen weiterverarbeitet werden.

Zur weiteren Verarbeitung der Daten wurde R [R10] verwendet. In R wurden die Variablen *zentriert* und *skaliert*, so dass für jede Variable der Mittelwert 0 und die Standardabweichung 1 ist.

3 Analyse der unklassifizierten Daten

Die 58 beobachteten Variablen sind stark korreliert. In Abb. 1 ist eine graphische Repräsentation der Korrelationsmatrix, ein *Corrgram* [Fr02], abgebildet. Die Reihenfolge der Variablen in der Korrelationsmatrix entspricht der Reihenfolge in Tabelle 1. Die Schattierung der Felder repräsentiert den Pearson Korrelationskoeffizienten der beiden Variablen (Schwarz = 1, Weiß = 0).

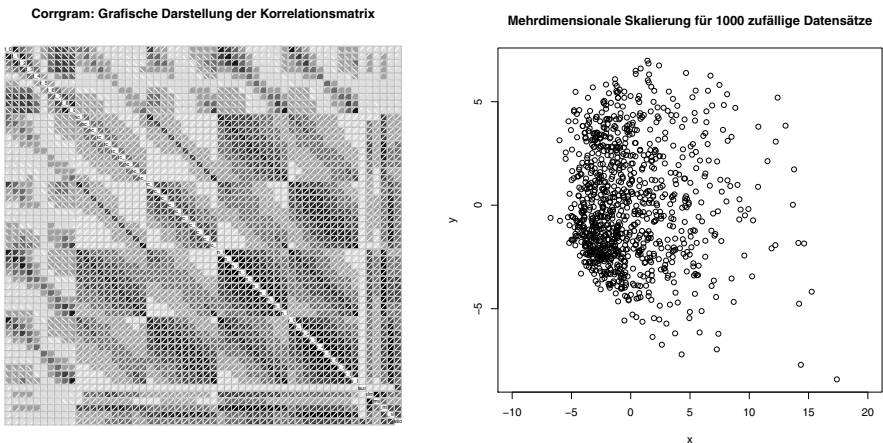


Abbildung 1: Links: Corrgram; Rechts: Multidimensionale Skalierung

Im Plot der multidimensionalen Skalierung für 1000 zufällig gewählte Datensätze ist keine Clusterstruktur erkennbar. Der *k-means* Algorithmus bestimmte für $k \in [2, 30]$ nur künstliche Cluster mit einer durchschnittlichen Silhouttenbreite (*average silhouette width* [Ro87]) kleiner 0,2. Auch ein dichtebasiertes Clustering mit DBSCAN [Es96] konnte keine natürliche Clusterstruktur in den Daten finden.

4 Erfassung von klassifizierten Daten

Zur weiteren Analyse wurden durch Studierende der TU Darmstadt Trainingsdatensätze erfasst. Hierzu sortierten die Studierenden unter den gleichen Bedingungen nach einem vorgegebenen Sortieralgorithmus. Insgesamt wurden 490 Trainingsdatensätze für die folgenden Sortieralgorithmen erfasst:

1. Selection sort von links nach rechts (Suche die Karte mit kleinstem Wert und tausche diese an den linken Rand des unsortierten Bereichs),
2. Selection sort von rechts nach links (Suche die Karte mit größtem Wert und tausche diese an den rechten Rand des unsortierten Bereichs),
3. Bubble sort von links nach rechts (Gehe das Feld von rechts nach links durch und tausche immer die kleinere Karte nach links),
4. Bubble sort von rechts nach links (Gehe das Feld von links nach rechts durch und tausche immer die größere Karte nach rechts),
5. Gnomesort von links nach rechts (Kann sowohl als Insertionsort, als auch als bidirektionaler Bubble sort betrachtet werden).

Die Trainingsdaten wurden zusammen mit den 4000 unklassifizierten Datensätzen auf Mittelwert 0 zentriert und auf Standardabweichung 1 skaliert.

Die Trainingsdaten sind innerhalb der Klassen nicht multivariat normal verteilt und haben unterschiedliche Dispersion.

5 Ergebnisse

Mit den klassifizierten Datensätzen wurden verschiedene Klassifikationsverfahren des maschinellen Lernens evaluiert. Hierzu wurden die Datensätze zunächst in eine Trainingsmenge (2/3) und eine Testmenge (1/3) aufgeteilt. Die jeweiligen Klassifikationsverfahren wurden ausschließlich mit den Daten der Trainingsmenge trainiert. Das so entstehende Vorhersagemodell wurde mit Hilfe der Testmenge auf Genauigkeit überprüft. Die Fehlerrate gibt für Trainings- und Testdaten den Anteil von falsch klassifizierten Datensätzen an.

Die Fehlerraten für die jeweiligen Verfahren wiesen in Abhängigkeit von der Verteilung der Datensätze auf beiden Mengen eine so starke Streuung auf, dass ein genauer Vergleich der Verfahren nicht möglich war. Um stabilere Fehlerraten zu erhalten wurden die Verfahren anschließend mit *10-facher stratifizierter Kreuzvalidierung* getestet. Hierzu wurden die Datensätze in 10 Mengen partitioniert, wobei der Anteil der Datensätze einer Klasse in einer Menge dem Anteil in der Gesamtmenge entspricht. Für jede dieser Mengen wurde jedes Verfahren mit den übrigen Mengen trainiert und mit den Daten dieser Menge evaluiert. Als Gesamtfehlerrate wurde das arithmetische Mittel über die 10 Einzelfehlerraten

<i>Verfahren</i>	<i>Fehlerrate Training</i>	<i>Fehlerrate Test</i>
Lineare Diskriminanzanalyse [VR02]	0,029	0,049
J48 Entscheidungsbaum [HBZ09, WF05] (Implementierung von Quinlans C4.5 Algorithmus[Qu93])	0,013	0,103
5 Nächste-Nachbarn-Klassifikation (knn) [We05]	0,057	0,098
Random Forest [LW02]	0,000	0,054
Multinomiale logistische Regression (multinom, nnet package [VR02])	0,000	0,141
L2 - Regularisierte multinomiale logistische Regression (LIBLINEAR) [Fan08, He10]	0,008	0,054
Support Vector Machine mit linearem Kernel[Di10]	0,022	0,060
Naiver Bayes-Klassifikator unter der Annahme multinomialer Normalverteilung [We05]	0,118	0,134
Naiver Bayes-Klassifikator mit Kerndichteschätzer	0,057	0,086
sparseLDA mit max. 35 Koeffizienten pro Diskriminante [CHE08, CI08]	0,037	0,050
Regularized Discriminant Analysis [Fr89, We05]	0,000	0,023

Tabelle 2: Vergleich der Fehlerraten für verschiedene Klassifikationsverfahren 10-fold cv

gebildet. Die Fehlerraten für eine Auswahl der getesteten Klassifikationsverfahren sind in Tabelle 2 abgebildet.

Auffällig ist das gute Ergebnis der Linearen Diskriminanzanalyse (LDA) [VR02], obwohl die Voraussetzungen, eine multivariate Normalverteilung mit gleicher Dispersion in den jeweiligen Klassen, nicht gegeben ist. Eine Quadratische Diskriminanzanalyse (QDA) [VR02] konnte wegen der Anzahl der Variablen nicht angewendet werden. Friedmans Regularized Discriminant Analysis (RDA) [Fr89, We05] erzielte die besten Ergebnisse.

Multinomiale logistische Regression erzeugte eine deutliche Überanpassung des Modells an die Trainingsdaten. Diese Überanpassung konnte mit einer L_2 -Norm Regularisierung [Fan08, He10] vermieden werden. Ähnliche Überanpassungen konnten für andere Verfahren beobachtet werden, die komplexere, nicht-lineare Entscheidungsgrenzen bestimmen. Verursacht wurden diese Überanpassungen durch die vergleichsweise große Anzahl der Variablen.

Sehr deutlich wird dieses Problem der hohen Dimensionalität im direkten Vergleich von LDA und QDA. Es sei $\mathbf{X} \in \mathbb{R}^{n \times p}$ die Matrix aller Datensätze mit n Datensätzen und p beobachteten Variablen. Für jede der k -Klassen $C_1, \dots, C_k$ sei n_i die Anzahl der Datensätze in dieser Klasse und $\mathbf{m}_i$ der Schwerpunkt (arithmetisches Mittel) der Klasse.

Die LDA ordnet einen neuen Datensatz $\mathbf{x}'$ derjenigen Klasse zu, deren Schwerpunkt $\mathbf{m}_i$ den geringsten Abstand zu $\mathbf{x}'$ hat. Der Abstand wird in der Mahalanobisdistanz

$$D_i(\mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{m}_i)^t \mathbf{S}_W^{-1} (\mathbf{x} - \mathbf{m}_i)}$$

gemessen. Wobei $\mathbf{S}_W$ die klassenübergreifende Kovarianzmatrix bezeichnet (vgl. [HTB94]).

$$\mathbf{S}_W = \sum_{i=1}^k \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^t$$

Die QDA unterscheidet sich von der LDA nur in der Distanzmessung. Statt einer klassenübergreifenden Kovarianzmatrix wird für jede Klasse eine klassenspezifische Kovarianzmatrix verwendet. Im vorliegenden Fall würde dann die 56×56 Kovarianzmatrix aus nur noch ca. 40 Datensätzen geschätzt werden. Es ist offensichtlich, dass eine so geschätzte Kovarianzmatrix keine zufriedenstellenden statistischen Eigenschaften besitzt. Probleme aufgrund der Dimensionalität können mit den folgenden Verfahren vermindert werden:

1. Vorherige Variablenauswahl (z. B. nach der Korrelation mit der Klassenzugehörigkeit),
2. Bestimmen von latenten Variablen mit Hilfe von Hauptkomponentenanalyse (Principal component analysis) [Ho33, Jo02],
3. Wahl von Klassifikationsverfahren, die die Variablen während der Trainingsphase auswählen (z. B. Entscheidungsbäume [Qu93]),
4. Wahl von regularisierten Klassifikationsverfahren. Erwähnenswert sind insbesondere die aus der Regression bekannten L_1 [Ti96] und L_2 Norm Regularisierungen [ZH05], die in den letzten Jahren für verschiedene Klassifikationsverfahren genutzt wurden (s. u.a. [Yu10]).

Im vorliegenden Fall, haben sich insbesondere die regularisierten Verfahren bewährt, die auch bei wenigen Trainingsdatensätzen akkurat (Fehlerrate 5 – 7%) klassifizierten. Erwähnenswert ist, dass eine Regularisierung mit L_1 Norm einige Koeffizienten auf 0 schrumpft, während Regularisierung mit L_2 Norm die Koeffizienten des Modells zwar schrumpft, diese aber nicht 0 werden. Stattdessen schrumpft die Regularisierung mit L_2 Norm Koeffizienten korrelierter Variablen auf ähnliche Werte (vgl. [ZH05])

Friedmans RDA [Fr89] ist eine spezielle Form der Regularisierung, die einen Kompromiss zwischen QDA und LDA darstellt. Die Kovarianzmatrizen für die jeweiligen Klassen werden durch die klassenübergreifende Kovarianzmatrix und ein Vielfaches der Einheitsmatrix regularisiert. Die geringere Fehlerrate der RDA gegenüber der LDA erklärt sich aus der beobachteten unterschiedlichen Dispersion der Daten in den Klassen.

5.1 Dimensionsreduktion mit Fishers linearer Diskriminante

Eine alternative Formulierung der LDA geht auf Fisher [Fi36] zurück. Gesucht ist eine lineare Projektion $\mathbf{W}^t$, die die Datensätze in einen $k - 1$ -dimensionalen Raum (k ist die Anzahl der Klassen) projiziert, so dass die Schwerpunkte der Klassen möglichst weit auseinanderliegen und die klassenübergreifende Kovarianzmatrix im projizierten Raum der Einheitsmatrix entspricht. Diese Projektion kann durch Eigenwertzerlegung recht einfach bestimmt werden (vgl. [VR02]). Die Spalten von $\mathbf{W}$ nennt man Diskriminanten (discriminant directions).

Der durch diese Projektion entstehende $k - 1$ -dimensionale Raum enthält alle nötigen Abstandsinformationen der Datensätze. Der euklidische Abstand im Bild von $\mathbf{W}^t$ entspricht der Mahalanobisdistanz der einzelnen Datensätze. Die Dimension kann noch weiter reduziert werden, in dem nur die ersten $k' < k - 1$ Diskriminanten verwendet werden. Dieses Verfahren ist als *Reduced rank LDA* bekannt und kann zu einer höheren Genauigkeit führen (vgl. [HTB94]).

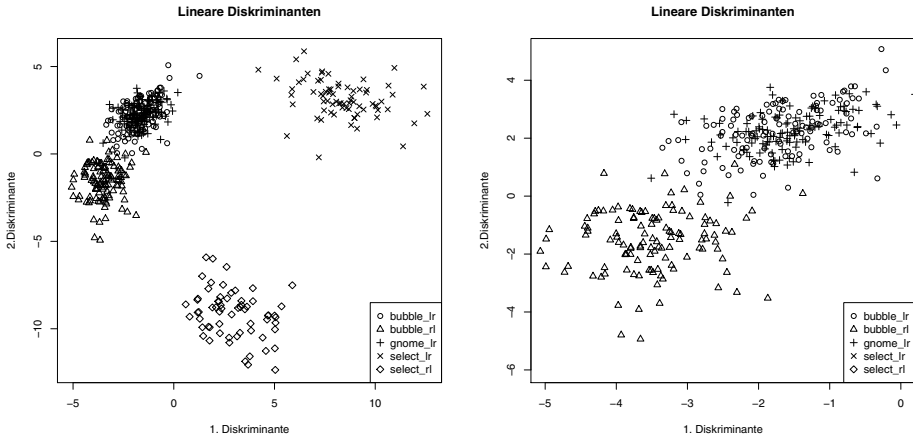


Abbildung 2: Projektion der klassifizierten Daten auf die ersten beiden Diskriminanten

Die so entstehende Projektion kann nicht nur für weitere Analysen der Daten verwendet, sondern auch für eine geeignete grafische Darstellung der Datensätze benutzt werden. So enthalten die ersten zwei Diskriminanten für die klassifizierten Daten über 90 % der zur Klassifikation nötigen Informationen. Die Projektion der Daten auf die ersten zwei Diskriminanten sind in Abb. 2 abgebildet. Die beiden Selectionsort Varianten werden optimal getrennt (s. Abb 2 links). Die Überlagerungen zwischen Gnomesort und Bubblesort (s. Abb. 2 rechts) sind durch die große Ähnlichkeit der Verfahren zu erklären. Mit den übrigen zwei Diskriminanten gelingt die Trennung besser, allerdings nicht perfekt. Die Fehlerrate (s. Tab. 2) der LDA und anderer Verfahren erklärt sich durch diese Überdeckungen.

6 Zusammenfassung und Ausblick

Handlungsfolgen beim Sortieren von Karten können mit Hilfe von verschiedenen Verfahren des maschinellen Lernens mit hoher Genauigkeit (>90 %) konkreten Sortierstrategien zugeordnet werden. Die Lineare Diskriminanzanalyse (LDA) erreichte eine gute Testfehlerrate von 5 %, obwohl die Voraussetzung einer möglichst gleichen Dispersion in den Klassen nicht gegeben war. Die geringste Fehlerrate erreichte Friedmans Regularized Discriminant Analysis [Fr89] eine regularisierte Erweiterung der LDA.

LDA kann nicht nur zur Klassifizierung, sondern auch zur Dimensionsreduktion genutzt werden. Die Projektion in den niederdimensionalen Raum kann nicht nur zur grafischen Darstellung, sondern auch zur weiteren Analyse und Verarbeitung der Daten verwendet werden. Diese Projektion ist ein Kompromiss zwischen Bias und Varianz. Im projizierten Raum sind die unterschiedlichen Klassen durch lineare Entscheidungsgrenzen zu 95 % trennbar.

Gegenstand weiterer Untersuchungen sind die erfassten unklassifizierten Datensätze. Erste Analysen mit Clusteringverfahren konnten keine natürliche Clusterstruktur identifizieren. Mögliche Ursachen hierfür sind verrauschte Daten, überlappende Klassen und ein degenerierendes Distanzmaß in höheren Dimensionen.

In weiteren Untersuchungen sollen mit Verfahren der Novelty Detection bisher nicht klassifizierte Sortierstrategien entdeckt werden. Der Klassifikator soll im Anschluß, um diese neuen Sortierstrategien erweitert werden. Mit diesem Klassifikator können anschließend die Fragestellungen zur Veränderung der Strategien eines Schülers untersucht werden.

Literaturverzeichnis

- [CHE08] Line Clemmensen, Trevor Hastie und Bjarne Ersbøll. Sparse discriminant analysis. Bericht, Technical University of Denmark, Lyngby, June 2008.
- [Cl08] Line Clemmensen. Sparse discriminant analysis (sparseLDA) software in R, may 2008.
- [Di10] Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, und Andreas Weingessel. *e1071: Misc Functions of the Department of Statistics (e1071)*, TU Wien, 2010. R package version 1.5-24.
- [Du35] Konrad Duncker. *Zur Psychologie des produktiven Denkens*. Julius Springer, 1935.
- [Es96] Martin Ester, Hans P. Kriegel, Jörg Sander und Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Evangelos Simoudis, Jiawei Han und Usama Fayyad, Hrsg., *Second International Conference on Knowledge Discovery and Data Mining*, Seiten 226–231, Portland, Oregon, 1996. AAAI Press.
- [Fan08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang und Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874, August 2008.
- [Fi36] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugen.*, 7:179–188, 1936.
- [Fr89] Jerome H. Friedman. Regularized Discriminant Analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.
- [Fr02] Michael Friendly. Corrgrams: Exploratory Displays for Correlation Matrices. *The American Statistician*, 56(4):316–324, 2002.

- [Fu03] Joachim Funke. *Problemlösendes Denken*. Kohlhammer, Stuttgart, 2003.
- [HBZ09] Kurt Hornik, Christian Buchta und Achim Zeileis. Open-Source Machine Learning: R Meets Weka. *Computational Statistics*, 24(2):225–232, 2009.
- [He10] Thibault Helleputte. *LiblineaR: Linear Predictive Models Based On The Liblinear C/C++ Library*, 2010. R package version 1.51-3.
- [Ho33] Harold Hotelling. Analysis of complex statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, September 1933.
- [HTB94] Trevor Hastie, Robert Tibshirani und Andreas Buja. Flexible Discriminant Analysis by Optimal Scoring. *Journal of the American Statistical Association*, 89:1255–1270, December 1994.
- [Jo02] Ian T. Jolliffe. *Principal Component Analysis*. Springer, 2nd. Auflage, October 2002.
- [Ki10a] Ulrich Kiesmüller. Automatisierte, prozessbegleitende Identifizierung der Problemlösestrategien Lernender unter Verwendung von Mustererkennungsmethoden. In *Didaktik der Informatik - Möglichkeiten empirischer Forschungsmethoden und Perspektiven der Fachdidaktik.*, Seiten 93–104, 2010.
- [Ki10b] Ulrich Kiesmueller, Sebastian Sossalla, Torsten Brinda und Korbinian Riedhammer. Online identification of learner problem solving strategies using pattern recognition methods. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, ITiCSE '10, Seiten 274–278, New York, NY, USA, 2010. ACM.
- [LW02] Andy Liaw und Matthew Wiener. Classification and Regression by randomForest. *R News*, 2(3):18–22, 2002.
- [MML08] Scott McQuiggan, Bradford Mott und James Lester. Modeling self-efficacy in intelligent tutoring systems: An inductive approach. *User Modeling and User-Adapted Interaction*, 18(1):81–123, February 2008.
- [NS72] Allen Newell und Herbert A. Simon. *Human Problem Solving*. Prentice-Hall, Inc., 1972.
- [Pu07] Hermann Puhlmann et al. Grundsätze und Standards für die Informatik in der Schule. *LOG IN*, 146/147, 2007.
- [Qu93] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [R10] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.
- [RJJ10] V. G. Renumul, Dharanipragada Janakiram und S. Jayaprakash. Identification of Cognitive Processes of Effective and Ineffective Students During Computer Programming. *Trans. Comput. Educ.*, 10, August 2010.
- [Ro87] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, November 1987.
- [SS04] Sigrid Schubert und Andreas Schwill. *Didaktik der Informatik*. Spektrum Akademischer Verlag, 1., Auflage, 2004.
- [Ti96] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.
- [VR02] W. N. Venables und B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth. Auflage, 2002. ISBN 0-387-95457-0.
- [WF05] Ian H. Witten und Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd. Auflage, 2005.
- [WG09] Christian Wach und Jens Gallenbacher. Spielend Sortieren mit Las Vegas Cardsort. In Bernhard Koerber, Hrsg., *INFOS*, Jgg. 156 of LNI, Seiten 256–267. GI, 2009.
- [We05] Claus Weihs, Uwe Ligges, Karsten Luebke und Nils Raabe. klaR Analyzing German Business Cycles. In L. Schmidt-Thieme, Hrsg., *Data Analysis and Decision Support*, Seiten 335–343, Berlin, 2005. Springer-Verlag.
- [Yu10] Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh und Chih-Jen Lin. A Comparison of Optimization Methods and Software for Large-scale L1-regularized Linear Classification. *Journal of Machine Learning Research*, Seiten 3183–3234, 2010.
- [ZH05] Hui Zou und Trevor Hastie. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.

Das Lernfeldkonzept in den Lehrplänen der IT-Berufe – Vorstudie zur schülerseitigen Akzeptanz und Umsetzbarkeit von selbstgesteuerten Lerneinheiten im Lernfeld „Entwickeln und Bereitstellen von Anwendungssystemen“

Simone Opel

Didaktik der Informatik
Friedrich-Alexander-Universität Erlangen-Nürnberg
Martensstraße 3
91058 Erlangen
simone.opel@informatik.uni-erlangen.de

Abstract: Das Konzept der Lernfelder wird in den IT-Berufen an vielen Schulen zum aktuellen Zeitpunkt kaum in handlungsorientierte Lernsituationen umgesetzt. Aus diesem Grund streben wir an, für die Berufe des IT-Bereichs durchgängige Lernsituationen für verschiedene Lernfelder unter Verwendung unterschiedlicher Unterrichtsmittel zu entwickeln und zu evaluieren. Im ersten Schritt werden verschiedene Unterrichtsmittel hinsichtlich ihre Tauglichkeit und Akzeptanz im Berufsschuleinsatz erprobt. Die vorliegende Vorstudie untersucht, ob die Roboter der Lego Mindstorms-Reihe im Unterricht an Berufsschulen im Lernfeld „Entwickeln und Bereitstellen von Anwendungssystemen“ erfolgreich eingesetzt werden können. Die Ergebnisse des Unterrichtsversuchs zeigen, dass sowohl hinsichtlich des Interesses und der Motivation der Schülerinnen und Schüler als auch bezüglich des Wissenszuwachses das Unterrichtsmittel Lego Mindstorms an der Berufsschule sinnvoll verwendet werden kann.

1 Motivation

1.1 Das Lernfeldkonzept in der beruflichen Bildung

In jedem Jahr beginnt in Deutschland rund eine Million junger Menschen ein Studium oder eine berufliche Ausbildung. Öffentliche Beachtung erfährt insbesondere die Hochschulbildung, obwohl die Anzahl derer, die eine berufliche Ausbildung aufnehmen, die Anzahl der Studienanfänger bei weitem überschreitet (Studienanfänger Wintersemester 2010/2011: 443 000 [SBA11], neu abgeschlossene Ausbildungsverträge zum Herbst 2010: 560 000 [BMBF11]). Es ist daher wünschenswert, dass der Lernort Berufsschule und die damit verbundenen Lernortkooperationen und Besonderheiten der dualen Ausbildung mehr Beachtung innerhalb der Bildungsforschung finden.

So liegt im Gegensatz zu allgemeinbildenden Schulen der Bildungsauftrag der Berufsschule gemäß der Rahmenvereinbarung über die Berufsschule [KMK91] in der Vermittlung beruflicher und allgemeinbildender Lerninhalte „unter besonderer Berücksichtigung der Anforderungen der Berufsausbildung“. In den Ausführungen zum Bildungs- und Erziehungsauftrag in den Lehrplänen und Lehrplanrichtlinien wird insbesondere auf die Vermittlung von Berufsfähigkeit, Verbindung von Fachkompetenz mit allgemeinen Fähigkeiten humaner und sozialer Art und die Hinführung zu beruflicher Flexibilität der Auszubildenden verwiesen. Als Mittel zur Erreichung dieser Ziele wird eine Ausrichtung des Unterrichts an handlungsorientierter Pädagogik genannt.

Der bis heute auch an vielen beruflichen Schulen durchgeführte Unterricht in Fächern, die nach rein fachsystematischen Gesichtspunkten und nicht nach Lernfeldkriterien definiert wurden, eignet sich nur begrenzt zur Umsetzung dieser didaktischen Vorgaben, da dieses Konzept die Themen fachlich eingrenzt und nur bedingt eine ganzheitliche Betrachtung der einzelnen Handlungen erlaubt. Somit ist es folgerichtig, dass mit der Einführung der neuen IT-Berufe im Juli 1997 eine formell konsequente Umstellung der Lehrpläne weg von der Fachsystematik hin zum Lernfeldkonzept betrieben wurde. Die Kultusministerkonferenz [KMK07] definiert ein Lernfeld als „durch Ziel, Inhalte und Zeitrichtwerte beschriebene thematische Einheiten, die an beruflichen Aufgabenstellungen und Handlungsfeldern orientiert sind und den Arbeits- und Geschäftsprozess reflektieren“. Diese Formulierung beinhaltet neben den fachpraktischen Fragestellungen auch fachwissenschaftliche Themen. Für Bader [Ba98] sind daher Lernfelder „nicht einfach in den Unterricht abgebildete berufliche Handlungsfelder, sondern didaktisch-methodische Konstrukte, die durch Reflexion und Rekonstruktion beruflichen Handelns gewonnen werden. Lernfelder sind didaktisch begründete und für den Unterricht aufbereitete Handlungsfelder“. Es ist Aufgabe der Lernortkooperationspartner, Lernfelder durch didaktische Reflexion zu konkretisieren und hieraus greifbare Lernsituationen zu formulieren [Ba03], die neben Anwendungsbezügen auch Theoriewissen vermitteln, da dies immer die Grundlage von Problemlösungsstrategien und Expertise sein muss [BS98].

Durch die Möglichkeiten zur Ausgestaltung, die diese Idee den Lehrenden und Lernenden geben kann, ist es nicht verwunderlich, dass zum heutigen Zeitpunkt das Lernfeldkonzept in einen wesentlichen Teil der Lehrpläne in verschiedenen Berufsfeldern Eingang gefunden hat. Dennoch erfährt das Konzept zumindest in den IT-Berufen im Bundesland Bayern und auch darüber hinaus nur zögerliche Umsetzung im schulischen Alltag. Ebenso selten finden sich in der Literatur Konzepte zur Umsetzung konkreter Lernsituationen eines Lernfeldes. Die wenigen dokumentierten Projekte und Konzepte betreffen in der Regel das Lernfeld „Entwickeln und Bereitstellen von Anwendungssystemen“ (z. B. bei Johlen [Jo02], Vocke & Woigk [VW05] oder Repp et al. [RZM07]). So sinnvoll und vorbildlich die jeweils beschriebenen Projekte auch sind, sie decken jeweils nur einen kleinen Bereich des Lehrplanes ab und dienen meist der Festigung und Vertiefung, nicht der Erarbeitung neuer Inhalte und sind in der Regel auch nicht an das entsprechende Lernfeld angepasst. Wünschenswert wären daher durchgängige, theoretisch fundierte Konzepte, die den gesamten Lehrplan oder zumindest ein Lernfeld vollständig umfassen.

1.2 Langeweile als möglicher Prädiktor der Passung eines Unterrichtsmittels

Im Gegensatz zu allgemeinbildenden Schulen sind Klassen an Berufsschulen sehr inhomogen hinsichtlich Alter und Vorausbildung. Das Durchschnittsalter eines Berufsschülers ist inzwischen auf 19 Jahre angestiegen, viele Schüler besitzen Hochschulreife und haben auf Grund ihrer gestiegenen Medienorientierung einen hohen Grad an Informiertheit bei gleichzeitig abnehmender Bereitschaft, diese Flut zu filtern und zu bewerten. Bei Themenbereichen, die für sie interessant sind, sind sie aber durchaus fähig, konzentriert zu arbeiten [RM04]. Eine Untersuchung, welche Unterrichtsmedien, -methoden und -prozesse für Berufsschüler interessant und motivierend sein können, sollte daher der erste Schritt bei der Erarbeitung von Umsetzungskonzepten verschiedener Lernfelder sein.

Schiefele & Pekrun [SP96] haben sich intensiv mit Lernstrategien und selbstgesteuertem Lernen auseinandergesetzt. Sie erarbeiteten hierzu ein Phasenmodell selbstregulierten Handelns, das neben externer Lernsteuerung durch situative Merkmale und die Lernumgebung auch die interne Lernsteuerung der Schüler in verschiedenen Phasen und Dimensionen enthält. Nach diesem Modell sind während der Vor- und Nachbereitung von Lernsituationen metakognitive Aspekte und überdauernde Motivationen von großer Bedeutung, dagegen spielen zum Zeitpunkt der Durchführung zusätzlich gegenwartsorientierte Emotionen wie Langeweile, aktuelles Interesse oder motivationale Tendenzen eine große Rolle, so dass auf ihnen das Augenmerk bei der Evaluierung der prinzipiellen Eignung einer Methode für den Berufsschulunterricht liegt.

Viele Studien beschäftigen sich mit dem Aspekt der Langeweile im Unterricht. Pekrun definiert Langeweile als negative, aufgabenbezogene, während der Tätigkeit erlebte und desaktivierende Emotion [Pe98], Todt beschreibt Langeweile als Gegenteil von Interessiertheit [To90], und auch einige Studien (u. a. Pekrun & Hoffman [PH99] und Lohrmann [Lo08]) zeigen empirisch hohe negative Korrelationen zwischen Langeweile und Interessiertheit. Eindeutige Auslöser von Langeweile sind nicht empirisch gesichert, für die zeitstabile Form der Langeweile als Disposition und Bereitschaft, diese zu empfinden, fanden Studien überwiegend Hinweise auf Überforderung als Auslöser [GFH06], nicht jedoch schulische Leistungen. Es stellt sich als vielversprechend dar, bei der Evaluierung von Lernsituationen die Langeweile der Schüler neben anderen Skalen wie Selbstkonzept oder Motivation zu erfassen, da bei Langeweile im Gegensatz zu anderen Skalen keine Korrelationen mit den Schulleistungen (z. B. Dickhäuser & Stinsmeier-Pelster [DS03] oder Todt [To00]) zu erwarten sind.

1.3 Forschungsfragen

Ein Unterrichtsmittel, das in den letzten Jahren im Informatikunterricht an allgemeinbildenden Schulen vermehrt Einzug gehalten hat, sind die Roboter der Reihe „Lego Mindstorms“. Ihre Einsetzbarkeit in verschiedenen Szenarien und Jahrgangsstufen wurde – allerdings in der Regel unter Verwendung ikonischer Entwicklungsumgebungen – bereits gezeigt (z. B. Weber & Wiesner [WW09]), so dass es sich anbietet, sie auch bei der Konzeption von Lernsituationen für den Unterricht im IT-Bereich von Berufsschulen zu berücksichtigen. Hieraus ergeben sich zwei Forschungsfragen. Als erstes stellt sich die Frage, ob der Lernzuwachs der Schüler beim Einsatz von „Mindstorms“ mindestens

dem entspricht, der mit „klassischen“ Unterrichtsmitteln zu erwarten wäre. Die zweite Forschungsfrage beleuchtet die motivationale Eignung des Unterrichtsmittels. Zu untersuchen ist, ob der Einsatz von „Lego Mindstorms“ die situative Langeweile der Schüler am Unterrichtsstoff vermindert. Es wird erwartet, dass „Mindstorms“ auch im handlungsorientierten Unterricht an Berufsschulen einen Lernzuwachs erzielen, der dem mit konventionellen Mitteln geführten Unterricht nicht nachsteht, allerdings wird gleichzeitig eine geringe Langeweile der Schüler im Unterricht erwartet.

2 Methode

Die an dieser Stelle vorgestellte Vorstudie verfolgt das Ziel, die Eignung der Mindstorms-Roboter für den Unterricht in IT-Klassen von Berufsschulen zu ermitteln. Die motivationale Tauglichkeit wird an Hand der von den Schülern berichteten situativen schulischen Langeweile, die fachliche Eignung durch eine entsprechende Leistungserhebung, unterstützt durch eine Kontrollgruppe, ermittelt.

Stichprobe: Um die Akzeptanz des Unterrichtsmittels „Mindstorms“ in der Vorstudie zu testen, wurden zwei Berufsschulklassen ($N = 60$) des ersten Ausbildungsjahres zum Fachinformatiker Systemintegration ausgewählt. Der Anteil der Schülerinnen beträgt 8,3% ($n = 5$). Das Alter der Schülerinnen und Schüler liegt zwischen 16 und 28 Jahren, der Median ist bei 20 Jahren. Auch hinsichtlich des Schulabschlusses sind die beiden Klassen sehr inhomogen: Drei Schüler haben einen qualifizierenden Hauptschulabschluss (5%), 24 Schüler mittlere Reife (40%), 11 Schüler besitzen Fachabitur (18%), allgemeine oder fachgebundene Hochschulreife haben 21 (35%) Schüler, ein Schüler traf keine Aussage. Die meisten Schüler hatten vor der Ausbildung keine ($n = 23$; 38%) oder geringe ($n = 18$; 30%) Programmierkenntnisse, nur wenige Schüler gaben an, mittlere ($n = 11$; 18%) oder gute ($n = 7$; 12%) Vorkenntnisse zu besitzen. Ein Schüler machte keine Angaben. Die Mehrzahl der Schülerinnen und Schüler ($n = 51$; 86%) ist im Betrieb nie oder nur selten im Bereich Programmierung oder Softwareentwicklung eingesetzt. Von den restlichen Schülern berichten sechs (10%), häufiger in diesem Bereich zu arbeiten, ein Schüler (2%) sogar regelmäßig. Ein Schüler macht keine Angaben.

Beide Klassen hatten bis zum Beginn der Vorstudie zwei Unterrichtsblöcke zu je zwei Wochen, während denen sie im Fach Anwendungsentwicklung und Programmierung (AP; Fachliche Umsetzung des Lernfeldes „Entwickeln und Bereitstellen von Anwendungssystemen“) grundlegende Algorithmen und die Grundprinzipien der strukturierten Programmierung in C bis zur Verwendung von Verzweigungen erlernten und einübten. Die eigentliche Vorstudie mit dem Inhalt „Kontrollstrukturen: Wiederholungen und Schleifen“ umfasst acht Unterrichtsstunden, die innerhalb der zweiten Woche des dritten Unterrichtsblocks stattfanden. Die erste der beiden Klassen wurde zur Mindstorms-Gruppe gewählt, die zweite Klasse dient als Kontrollgruppe und erhielt Materialien, die den bisherigen Unterrichtsmitteln angepasst sind und auf bekannten Werkzeugen und Entwicklungsumgebungen basieren. Aus der Mindstorms-Gruppe geben nur zwei (3.4%) an, jemals mit diesen Robotern gearbeitet zu haben.

Instrument zur Erfassung der Langeweile: Zur Erfassung der Langeweile wurde eine angepasste Version der Langeweile-Itemskala mit fünfstufigem Antwortformat (von „nie“ (1) bis „immer“ (5)) von Sparfeldt et al. [SBS09] verwendet. Die Skala umfasst 14 Items, die sprachlich an das Fach Anwendungsentwicklung und Programmierung (AP) angepasst wurden. Beispielitems sind „Während der letzten Stunden AP war ich mit den Gedanken ganz woanders“ oder „Aus Langeweile schaltete ich im AP-Unterricht ab“. Dieser Fragebogen wurde vor Beginn der Studie ($\alpha = .957$) und nach Abschluss der Vorstudie ($\alpha = .974$) erfasst und zeigt trotz der Modifikationen sehr gute interne Konsistenzen.

Durchführung: Vor Studienbeginn wurde in beiden Gruppen das Vorwissen durch einen eher schulischen Prüfung ähnlichen Test erfasst, der sowohl programmierlogisches als auch programmiersprachliches Wissen und Fertigkeiten abfragt. Der Test umfasst fünf frei zu beantwortende, voneinander unabhängige Problemstellungen:

Um den Stand des bisherigen programmiersprachlichen Wissens zu erheben, sollten die Schüler in einem vorgegebenen Struktogramm, das aus einer doppelt verschachtelten Verzweigung bestand, die notwendigen Variablen ermitteln und für diese passende Datentypen vorschlagen. Außerdem war die erste Verzweigung als Ausdruck in C/C++ zu formulieren. Um das prinzipielle Verständnis der zu Grunde liegenden Logik zu testen, bestanden die weiteren Aufgaben darin, Teile des Struktogrammes in Worten zu beschreiben, Fälle der Grafik richtig zuzuordnen und das Struktogramm zu erweitern.

Am Ende der Studie erhielten beide Gruppen neben dem Fragebogen zur Langeweile einen in zwei Bereiche gegliederten Test (im Folgenden als „Nachtest“ bezeichnet), der in der ersten Hälfte den Vorwissenstest (Nachtest, Teil 1) repliziert. Die zweite Hälfte (Nachtest, Teil2) besteht aus sieben Problemstellungen zum neu erarbeiteten Wissen über Schleifen, die ebenfalls sowohl allgemeine programmierlogische als auch sprachlogische Aspekte beinhalten:

Sprachlogisches Wissen wurde dadurch erfasst, dass die Schüler die verbale Beschreibung einer Problemstellung einmal als kopfgesteuerte und einmal als Zählschleife umsetzen sollten. Durch die Analyse vorgegebener Struktogramme mit verschachtelten Schleifen war es möglich, allgemeine programmierlogische Zusammenhänge bei den Schülern abzufragen.

Zusätzlich wurde die Mindstorms-Gruppe mit zwei Fragen im Sinne der Theory of planned Behaviour von Ajzen [AF75] nach Ihrer Einstellung zum Einsatz des Unterrichtsmittels „Mindstorms“ gefragt („Was spricht aus Ihrer Sicht für bzw. gegen AP-Unterricht mit den Mindstorms?“).

Da die Vorstudie nur eine kleine Stichprobengröße besitzt, wurde zur weitgehenden Elimination der direkten Einflüsse der Lehrkräfte ein Leittext-gestütztes Verfahren entwickelt. Die Schüler erhielten abgestimmt auf ihr individuelles Arbeitstempo fünf Einzelkapitel. Diese Kapitel – die hinsichtlich des Aufbaus für beide Gruppen analog gestaltet sind – beginnen immer mit Wiederholungsfragen zum vorherigen Kapitel, anschließend werden die neuen Lerninhalte basierend auf Problemen der Alltagswelt erarbeitet und in Übungen vertieft. Zur Kontrolle und Hilfestellung standen den Schülern zu jeder Zeit Tutorials und Lösungsvorschläge mit Erläuterungen zur Verfügung. Die Mindstorms-Gruppe verwendete als Entwicklungsumgebung BrixCC und die Sprache

NXC (NoteXactlyC), beides war ihnen bis zu diesem Zeitpunkt unbekannt. Die Schüler der Kontrollgruppe benützten wie bisher eine integrierte Entwicklungsumgebung für C und C++. Die Schüler beider Gruppen konnten während der gesamten Stundensequenz frei wählen, ob sie Einzel- oder Partnerarbeit bevorzugen.

3 Ergebnisse

Langeweile im Unterricht: Die von den beiden Gruppen berichtete Langeweile im Unterricht (Tabelle 1) unterscheidet sich sowohl hinsichtlich ihrer Mittelwerte (*M*) als auch der Standardabweichungen (*SD*). Während bei der Mindstorms-Gruppe die Langeweile während der Studie als geringer als zuvor berichtet wird, steigt sie bei der Kontrollgruppe während der Durchführung der Studie sogar an.

	MINDSTORMS- GRUPPE (<i>N</i> = 28)	KONTROLL- GRUPPE (<i>N</i> = 27)	T-TEST	
			<i>T</i>	<i>p</i>
VORABBEFRAGUNG:	<i>M</i> = 1.80 – <i>SD</i> = 0.79	<i>M</i> = 2.16 – <i>SD</i> = 0.95	-1.86	.069
BEFRAGUNG ZUR STUDIE:	<i>M</i> = 1.29 – <i>SD</i> = 0.49	<i>M</i> = 2.39 – <i>SD</i> = 0.98	-5.47	.000

Tabelle 1 Deskriptive Statistiken zur Skala der Langeweile

Überprüft man die Medianwerte der beiden Gruppen zu beiden Zeitpunkten mittels Test nach Neyman-Pearson auf einem Signifikanzniveau von $\alpha = .05$, wird die Hypothese, die Mediane beider Gruppen in der Vorabbefragung seien gleich, beibehalten ($p = .234$), die Hypothese, dass die Mediane bei der Befragung zur Studie (Zeitpunkt des „Nachtest“) ebenfalls gleich sind, aber abgelehnt ($p < .001$). Das Ergebnis, dass sich die Aussagen hinsichtlich der Langeweile im bisherigen Unterricht AP zwischen beiden Gruppen nicht signifikant unterscheiden, wohl aber hinsichtlich der Langeweile während der Studie, wird durch einen t-Test für unabhängige Stichproben bestätigt.

Dass die Schüler der Mindstorms-Gruppe den Unterrichtseinsatz der Roboter an sich als weitgehend positiv beurteilen, erkennt man auch an den Antworten der Fragen zu deren Einsatz im Unterricht. Obwohl die Formulierung auf Antworten zur Einstellung zielt, betreffen nur 49% ($n = 32$) der Antworten diesen Bereich, der Rest entstammt der wahrgenommenen Verhaltenskontrolle (Kontrollierbarkeit: 14 Aussagen; Selbstwirksamkeit: 19 Aussagen). Während die Aussagen zur Einstellung (Beispiele: „hat Spaß gemacht“, „war interessant“, „abwechslungsreich“) und Selbstwirksamkeit (Beispiele: „man lernt Programmierlogik“, „ich sehe die Erfolge“, „Funktion optisch sichtbar“, aber auch „wenig Lerneffekt“) mit je einer Ausnahme positiv sind, dominieren hinsichtlich der Kontrollierbarkeit die negativen Bewertungen („zu wenig Kontrolle durch den Lehrer“, „Dokumentation“, „es wäre besser, wenn jeder einen eigenen Roboter zur Verfügung hätte“).

Wissenstest: Die Ergebnisse der Wissenstests in beiden Gruppen unterscheiden sich, wie in Tabelle 2 ausgeführt, nicht signifikant. Sowohl ein t-Test unabhängiger Stichproben als auch ein Test nach Neyman-Pearson ergibt, dass die beiden Stichproben der gleichen Grundgesamtheit entnommen sind, was durch die niedrigen Effektstärken noch bestätigt wird. Die beiden Klassen zeigten im Vorwissenstest ähnlichen Kenntnisstand, und auch der Zuwachs an neuem Wissen ist bei beiden Klassen ähnlich. Die Befürchtung vieler

Lehrkräfte, dass durch die Konzentration der Schüler auf das Unterrichtsmittel „Roboter“ die wesentlichen Inhalte verloren gehen würden, hat sich also nicht bewahrheitet.

	MINDSTORMS-GRUPPE (N = 29)		KONTROLLGRUPPE (N = 26)		EFFEKT- STÄRKE <i>d</i>	T-TEST-WERTE	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>		<i>T</i>	<i>p</i>
VORWISSENSTEST	10.59	2.82	9.46	3.14	0.14	1.59	.118
NACHTEST, TEIL 1	9.41	3.91	11.12	2.55	-0.14	-1.59	.069
NACHTEST, TEIL 2	6.41	4.24	7.75	3.22	-0.09	-1.30	.198
NACHTEST, GESAMT	15.83	7.03	18.87	5.25	-0.08	-1.80	.078

Tabelle 2 Deskriptive Statistiken der Wissenstests

Bivariate Korrelationen der erhobenen Daten: Erwartungsgemäß zeigt sich, dass die Langeweile im bisherigen Unterricht hoch mit der Langeweile während der Studie ($r = .74$; $p < .001$) korreliert. Das programmiertechnische Vorwissen zu Beginn der Ausbildung stellt einen Prädiktor für die berichtete Langeweile im bisherigen Unterricht dar ($r = .298$; $p = .035$), der Effekt ist allerdings bei getrennter Betrachtung beider Gruppen nicht mehr signifikant. Um eindeutige Ergebnisse zu erhalten, sollte daher die Stichprobengröße erhöht werden. In Tabelle 3 wird ersichtlich, dass sich bei gemeinsamer Betrachtung aller Schüler signifikante Korrelationswerte insbesondere zwischen dem Alter der Schüler und den Ergebnissen des Wissenstests am Ende der Untersuchung ergeben. Wie zu erwarten war, bestehen zwischen dem Schulabschluss und den Ergebnissen des Wissenstests hoch signifikante Abhängigkeiten, außerdem zwischen den Vorkenntnissen der Schüler zu Beginn der Ausbildung und den Testergebnissen. Die Tätigkeit im Ausbildungsbetrieb dagegen spielt kaum eine Rolle.

	VORWISSENSTEST	NACHTEST, TEIL 1	NACHTEST, TEIL 2	NACHTEST, GESAMT
ALTER	.23	.28*	.33*	.35*
SCHULABSCHLUSS	.26	.34*	.47**	.46**
VORKENNTNISSE	.36*	.31*	.45**	.43**
BETRIEBL. EINSATZ	.31*	.09	.15	.14
LANGEW. VORABB.	-.11	-.14	-.06	-.11
LANGEW. STUDIE	-.12	.02	-.08	-.04

Anmerkungen. * $p < .05$; ** $p < .01$

Tabelle 3 Korrelationen nach Pearson der erhobenen Daten (N = 54)

Bei getrennter Betrachtung beider Gruppen stellen die Vorkenntnisse bei der Mindstorms-Gruppe keine Prädiktoren für die Ergebnisse der Tests mehr dar, lediglich die Schulbildung korreliert mit dem Ergebnis des Nachtests Teil 2 ($r = .41$; $p = .040$) und das Alter mit dem Gesamtergebnis des Nachtests ($r = .41$; $p = .040$). Bei der Kontrollgruppe stellen die Schulbildung und die Vorkenntnisse Prädiktoren der Ergebnisse der Vorwissens- und Nachtests dar. Zusätzlich korrelieren das Alter ($r = .44$; $p = .028$) und die Tätigkeit im Betrieb ($r = .43$; $p = .030$) mit den Ergebnissen des Vorwissenstests. Hinsichtlich der berichteten Langeweile finden sich in der Gesamtstichprobe keine signifikanten Korrelationen mit den Ergebnissen der Wissenstests. Bei der Mindstorms-Gruppe ergeben sich Effekte der Langeweile im bisherigen Unterricht mit dem Ergebnis des Teil 1 des Nachtests ($r = -.47$; $p = .012$) und des gesamten Nachtests ($r = -.38$; $p = .044$). Dies repliziert teilweise das Ergebnis von Sparfeldt et al. [SBS09], die eben-

falls signifikante Korrelationen zwischen Langeweile und Rechentests bzw. Rechennoten fanden.

4 Diskussion

An der Berufsschule, die die Vorstudie durchführte, stellte dies den ersten Einsatz des Unterrichtsmittels „Mindstorms“ im IT-Bereich dar. Projekte und selbstgesteuerte Unterrichtseinheiten wurden bisher wie an vielen Schulen nur zur Vertiefung und Festigung des im lehrerzentrierten Unterrichts erarbeiteten Stoffes verwendet, so dass diese Art der Unterrichtsgestaltung für alle Beteiligten ungewohnt war. Daher verwundert die Aussage einer beteiligten Lehrkraft, dass „man sich überflüssig vorkommt, da die Schüler ja selbstständig arbeiten“ nicht besonders – vor allem, da für die Lehrkräfte die bei diesen Unterrichtskonzepten aufwendige Vorbereitung der Materialien entfallen ist.

Wie die Befragungen ergeben, wird der Einsatz der Roboter im Unterricht Anwendungsentwicklung von den Schülern als abwechslungsreich und interessant angesehen. Die häufig geäußerten Befürchtungen, dass Berufsschüler Mindstorms als nicht alters- und ausbildungsgerecht ansehen könnten, oder die Roboter von den eigentlichen Lerninhalten ablenken könnten, konnten hier entkräftet werden, da hinsichtlich des Wissenszuwachses beide Gruppen gleichmäßig profitieren konnten, die Studie sich aber für die Mindstorms-Gruppe nach eigenen Aussagen kurzweiliger und interessanter gestaltete.

Schwierig gerade für Schüler mit niedrigerem Schulabschluss gestaltete sich in der Kontrollgruppe nach deren Aussagen die selbstständige Arbeit mit den doch recht umfangreichen Leittexten, was auch die Korrelation zwischen Testergebnissen und Schulabschluss gerade in der Kontrollgruppe mit erklären könnte. Auffällig bei der Beobachtung des Schülerverhaltens war, dass in dieser Klasse bevorzugt alleine gearbeitet wurde. Zudem war ein Teil der Schüler mit der Methode der Leittextarbeit wenig vertraut, und so fielen auch die Hilfestellungen untereinander geringer aus.

Positiv zu bewerten ist, dass trotz des Einsatzes dieser für die Schüler ungewohnten Methode in der Mindstorms-Gruppe ein Zusammenhang zwischen Schulabschluss bzw. dem programmiertechnischen Vorwissen und den Ergebnissen der Nachtests kaum mehr erkennbar ist. Das deutet an, dass diese Art von Unterrichtsmittel gut für die in der Berufsschule immer wieder anzutreffenden sehr heterogenen Klassen geeignet ist. Die Ursachen des Lernerfolgs der Unterrichtssequenz scheinen sich von externen Faktoren wie Unterrichtsstil der Lehrkraft oder Vorausbildung hin zu internen und individuellen Faktoren wie Interesse und Motivation zu verlagern. Dies stellt aber zum momentanen Zeitpunkt lediglich eine Hypothese auf Basis nicht repräsentativer Gespräche mit beteiligten Schülern und Beobachtungen während der Arbeit der beiden Gruppen dar.

Limitationen: So ermutigend die Ergebnisse dieser Studie sind, beziehen sie sich doch auf eine sehr kleine Stichprobe einer einzelnen Schule, können also nicht repräsentativ für den gesamten Bereich der IT-Klassen an Berufsschulen gelten. Aus diesem Grund wurde auch darauf verzichtet, die Daten mehrebenenanalytisch auszuwerten, was in größeren Folgestudien in jedem Fall sinnvoll sein wird. Auch ist nicht auszuschließen, dass die Anwesen-

heit der Lehrkräfte bei den Befragungen bezüglich gefühlter Langeweile sozial erwünschte Antworten bei den Schülern gefördert hat. Als weiteres Problem hat sich herausgestellt, dass die Mindstorms-Gruppe die Studie in der letzten Woche vor den Weihnachtsferien durchführte und somit der Nachtest in der letzten Unterrichtsstunde stattfand, was merklich zu Unruhe und unkonzentriertem Arbeiten führte. Eine Verzerrung des Testergebnisses ist somit nicht auszuschließen. Auch können natürlich Novitäts-Effekte nicht ausgeschlossen werden, so dass hier noch Forschungsbedarf vorhanden ist.

5 Fazit

Auch wenn die hier vorgestellte Studie noch weit entfernt ist von einer vollständigen Lernsituation zur Abbildung eines Lernfeldes, zeigt sie doch, dass gerade an Berufsschulen handlungsorientierte Unterrichtssequenzen nicht nur aus der Durchführung von Vertiefungsprojekten oder einfachen Implementierungsübungen bestehen müssen, sondern sich im Gegenteil ein breites Spektrum an Unterrichtsmitteln zum variablen Einsatz anbietet, das noch viel Raum für weitere Entwicklungen in sich trägt. Nicht außer Acht lassen darf man aber die Intention der mit offenen Unterrichtsmitteln gestalteten Stundensequenzen – die Roboter sind nur ein Medium, nicht der Inhalt des Unterrichts. Zudem ist es schwierig, nur mit einem einzelnen Unterrichtsmittel das gesamte Lernfeld abzudecken, da sich nicht alle Lehrplaninhalte des Lernfeldes „Entwickeln und Bereitstellen von Anwendungssystemen“ für einen adäquaten Unterricht anbieten. Daher ist die sorgfältige Entwicklung entsprechender Konzepte und Materialien eine wichtige Aufgabe, um in der Zukunft sinnvolle Lernsituationen zu gestalten.

Nächste Schritte werden daher sein, weitere Unterrichtsmittel im Einsatz zu erproben und zu evaluieren. Parallel dazu müssen in Zusammenarbeit mit ausbildenden Betrieben relevante Handlungsfelder der betrieblichen Praxis aufgespürt werden. Aus den Ergebnissen dieser beiden Teilprojekte werden durch didaktische Reduktion und unter Einbeziehung aktueller Erkenntnisse der Lehr- und Lernforschung exemplarische Lernsituationen entwickelt und zu einem theoretisch fundierten Gesamtkonzept zusammengefasst, um so die Umsetzung des Lernfeldgedankens in den Berufen des IT-Bereichs zu fördern.

Literaturverzeichnis

- [AF75] Ajzen, I.; Fishbein, M.: Belief, attitude, intention, and behavior: An introduction to theory and research. Reading, MA: Addison-Wesley, 1975.
- [Ba03] Bader, R.: Lernfelder konstruieren – Lernsituationen entwickeln. In: Die berufsbildende Schule, 55/2003, 2003; S. 210-217.
- [BMBF11] Bundesministerium für Bildung und Forschung (BMBF): Berufsbildungsbericht 2011, Bonn, Berlin: Bundesministerium für Bildung und Forschung, 2011.
- [BS98] Bader, R., Schäfer, B.: Lernfelder gestalten. Vom komplexen Handlungsfeld zur didaktisch strukturierten Lernsituation. In: Die berufsbildende Schule, Nr. 50, Heft 7-8, 1998; S. 229-234.
- [DS03] Dickhäuser, O.; Stiensmeier-Pelster, J.: Wahrgenommene Lehrereinschätzungen und das Fähigkeitsselbstkonzept von Jungen und Mädchen in der Grundschule. Psychologie in Erziehung und Unterricht, 50, 2003; S. 182-190.

- [GFH06] Götz, T.; Frenzel, A. C. & Haag, L.: Ursachen von Langeweile im Unterricht. *Empirische Pädagogik*, 20, 2006. S. 113-134.
- [Jo02] Johlen, D.: Methodik der OOSE für Fachinformatiker nach dem Lernfeldansatz unter Einbeziehung der Lehrerfortbildung. In: *Forschungsbeiträge zur „Didaktik der Informatik“ – Theorie, Praxis, Evaluation, Proceedings 1. GI-Workshop DDI 02*, Witten-Bommerholz, Bonn: Köllen, 2002.
- [KMK07] Sekretariat der Ständigen Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland (KMK): Handreichungen für die Erarbeitung von Rahmenlehrplänen der Kultusministerkonferenz (KMK) für den berufsbezogenen Unterricht in der Berufsschule und ihre Abstimmung mit Ausbildungsordnungen des Bundes für anerkannte Ausbildungsberufe. Bonn: KMK Sekretariat der Ständigen Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland, 2007.
- [KMK91] Sekretariat der Ständigen Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland (KMK): Rahmenvereinbarung über die Berufsschule. Bonn: KMK Sekretariat der Ständigen Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland, 2001.
- [Lo08] Lohrmann, K.: *Langeweile im Unterricht*. Münster: Waxmann, 2008.
- [Pe98] Pekrun, R.: Schüleremotion und ihre Förderung: Ein blinder Fleck der Unterrichtsforschung. *Psychologie in Erziehung und Unterricht*, 44, 1998; S. 230-248.
- [PH99] Pekrun, R.; Hoffmann, H.: Lern- und Leistungsemotionen: Erste Befunde eines Forschungsprogramms. In M. Jerusalem, R. Pekrun (Hrsg.): *Emotion, Motivation und Leistung*. Göttingen: Hogrefe, 1999.
- [RM04] Richter, C.; Meyer, R.: *Lernsituationen gestalten – Berufsfeld Elektrotechnik*. Troisdorf: Bildungsverlag, 2004.
- [RZM07] Repp, S.; Ziegler, R.; Meinel, C.: Lernortkooperation in der IT-Ausbildung - Kompetenzentwicklung in Projekten. In: Schubert, Sigrid E. (Hrsg.): *Tagungsband der 12. GI-Fachtagung Informatik und Schule (Infos), Didaktik der Informatik in Theorie und Praxis (Infos 2007)*, Siegen, Bonn: Köllen, 2007.
- [SBA11] Statistisches Bundesamt: *Bildung und Kultur: Studierende an Hochschulen – Vorbericht – Wintersemester 2010/2011*. Wiesbaden: Statistisches Bundesamt, 2011.
- [SBS09] Sparfeldt, J. R.; Buch, S.; Schwarz, F., Jachmann, J.; Rost, D.H.: „Rechnen ist langweilig“ – Langeweile in Mathematik bei Grundschulern. *Psychologie in Erziehung und Unterricht*, 56, 2009; S. 16-26.
- [SP96] Schiefele, U.; Pekrun, R.: Psychologische Modelle des fremdgesteuerten und selbstgesteuerten Lernens. In F. E. Weinert (Hrsg.): *Psychologie des Lernens und der Instruktion. Enzyklopädie der Psychologie, D/I, Bd. 2.* Göttingen: Hogrefe, 1996; S. 249–278.
- [To00] Todt, E.: Geschlechtsspezifische Interessen – Entwicklungen und Möglichkeiten der Modifikation. *Empirische Pädagogik*, 14, 2000; S. 215-254.
- [To90] Todt, E.: Entwicklung des Interesses. In H. Hetzer, E. Todt, I. Seiffge-Krenke, R. Arbiner (Hrsg.): *Angewandte Entwicklungspsychologie des Kindes- und Jugendalters* (2. Aufl.), Heidelberg: Quelle & Mayer, 1990; S. 213-264.
- [VW05] Vocke, H.; Woigk, U.: Software- Engineering in der beruflichen Ausbildung- Simulation realer Projektsituationen. In: *Gesellschaft für Informatik (Hrsg.): Tagungsband zur 11. GI Fachtagung „Informatik und Schule – Infos 2005“*, Springer-Verlag, Berlin Heidelberg New York, 2005; S. 297-307.
- [WW09] Weber, M.; Wiesner, B.: Informatische Konzepte mit Robotern vermitteln – Ein Unterrichtsprojekt für die Sekundarstufe I. In: *Gesellschaft für Informatik (Hrsg.): Tagungsband zur GI-Fachtagung "Informatik und Schule 2009 (INFOS2009)"*, Bonn: Köllen, 2009, 2005; S. 109-120.

Ein genetischer Zugang zum Programmieren mit CGI-Skripten in Python

Jan Schuster

Institut für Didaktik der Mathematik und der Informatik
Goethe-Universität Frankfurt am Main
Senckenberganlage 9
60325 Frankfurt am Main
schuster@math.uni-frankfurt.de

Abstract: In diesem Beitrag wird das genetische Prinzip, das in anderen Fachdidaktiken schon lange eine wichtige Rolle spielt, auf die Informatikdidaktik übertragen. Wie genetischer Informatikunterricht aussehen kann und welche Vorteile dieser bietet, wird in einer Unterrichtseinheit zur Einführung in das Programmieren mit CGI-Skripten in Python gezeigt.

1 Einleitung

Genetisches Unterrichten ist in vielen Didaktiken ein anerkanntes Prinzip. In der Informatikdidaktik wird es allerdings kaum explizit thematisiert. Was das genetische Prinzip für den Informatikunterricht bedeuten kann und wie man es dort umsetzen kann, wird in diesem Beitrag ausgearbeitet.

Im ersten Abschnitt werden zunächst verschiedene Theorien zum Genetischen Prinzip (vorwiegend) aus der Mathematikdidaktik dargestellt, und anschließend wird diese auf die Informatik übertragen. Am Ende steht eine Liste von Kriterien für genetischen Informatikunterricht.

Im zweiten Teil wird am Beispiel der Einführung in das Programmieren mit CGI-Skripten in einem Grundkurs Informatik exemplarisch aufgezeigt, wie diese Prinzipien in die Planung einer Unterrichtseinheit einfließen können.

2 Das genetische Prinzip und seine Bedeutung für die Planung von Informatikunterricht

2.1 Das genetische Prinzip in der Mathematikdidaktik

Das Wort „genetisch“ kommt vom griechischen Wort γίνομαι (ginomai), das „werden, entstehen, geboren werden“ bedeutet. Dazu passend definiert Wittmann für den Mathematikunterricht, dass ein Unterricht genetisch heißt, wenn die Darstellung der Theorie „an natürlichen erkenntnistheoretischen Prozessen der Erschaffung und Anwendung von Mathematik ausgerichtet ist“ ([Wi81], S. 130). Dies bedeutet, dass die Schüler und Schülerinnen¹ Mathematik nicht als fertiges Konzept kennenlernen sollen, sondern den Schöpfungsprozess, der dazu geführt hat, noch einmal nachempfinden bzw. nachvollziehen sollen. Mathematik soll als etwas lebendiges, neu zu Entdeckendes, zu Erfindendes unterrichtet werden.

Der genetische Mathematikunterricht steht im Gegensatz zum deduktiven Stil der Hochschulmathematik, bei dem von Axiomen ausgehend die fertige Mathematik abgeleitet wird. Ein genetischer Unterricht startet beim Vorwissen der Schüler und baut dieses Schritt für Schritt aus.

Wittmann nennt folgende Merkmale, die genetischen Unterricht in der Mathematik (GMU) charakterisieren:

- (GMU1) „Anschluss an das Vorverständnis der Adressaten,
- (GMU2) Einbettung der Überlegungen in größere ganzheitliche Problemkontexte außerhalb oder innerhalb der Mathematik,
- (GMU3) Zulässigkeit einer informellen Einführung von Begriffen aus dem Kontext heraus,
- (GMU4) Hinführung zu strengen Überlegungen über intuitive und heuristische Ansätze,
- (GMU5) Durchgehende Motivation und Kontinuität,
- (GMU6) Während des Voranschreitens allmähliche Erweiterung des Gesichtskreises und entsprechende Standpunktverlagerungen“ ([Wi81], S. 131).

Namhafte Mitbegründer, Förderer und Vertreter der genetischen Methode waren Felix Klein, Otto Toeplitz, Hans Freudenthal, Anna Z. Krygowska, Alexander I. Wittenberg, Martin Wagenschein, Jean Piaget, Jerome S. Bruner; um nur einige zu nennen. Alle teilen die gemeinsame Überzeugung, „dass die Mathematik nur über den Prozess der Mathematisierung richtig verstanden und erlernt werden kann, nicht als Fertigfabrikat“ ([Wi81], S. 131).

¹ Im Sinne der besseren Lesbarkeit wird im Folgenden meist nur noch die männliche Form gebraucht um sowohl Schüler als auch Schülerinnen zu bezeichnen.

Besonders stark prägte Felix Klein das Genetische Prinzip. Seine Vorstellungen für den Mathematikunterricht entwickelte er insbesondere in seinem dreibändigen Werk „Elementarmathematik vom höheren Standpunkt aus“ [Kl24]: „...anschaulich und genetisch, d.h., das ganze Lehrgebäude wird auf Grund bekannter anschaulicher Dinge ganz allmählich von unten aufgebaut; hierin liegt ein scharf ausgeprägter Gegensatz gegen den meist auf Hochschulen üblichen logischen und systematischen Unterrichtsbetrieb.“ (S. 6)

Auch Wittenberg forderte für den Mathematikunterricht eine Wiederentdeckung der Mathematik von Anfang an. Dazu entwickelte er auf Grundlage des genetischen Prinzips seine Themenkreismethode (vgl. [Wi63]). Er fordert eine „Auswahl des Unterrichtsstoffes, die diesen als organische Verknüpfung einiger weniger, bedeutungsvoller, verhältnismäßig umfassender, um eine sehr geringe Zahl einfacher mathematischer Tatsachen angeordneter Themenkreise erscheinen lässt; als einen wohlgestalteten, überzeugenden, überschaubaren gedanklichen Bau“ ([Wi63], S. 141). Dabei sollen insbesondere die letzten Punkte den Schüler überzeugen und nicht den Mathematiker. Für den Unterricht fordert Wittenberg, die „Gegenstände des Unterrichts so zu organisieren, wie sie tatsächlich in einer sachgemäß fortschreitenden Untersuchung ursprünglich hätten erschlossen werden können“ ([Wi63], S. 145). Die Erarbeitung eines Themenkreises soll grundsätzlich in zwei Stufen erfolgen: „allmählich fortschreitende Entfaltung von einem einfachen Ausgangspunkt aus; sodann zusammenfassender Rückblick auf das Erarbeitete, das in der Rückschau als ein durchsichtiger, klar gegliederter, deutlich sachbezogener gedanklicher Bau gesehen werden muss“ ([Wi63], S.147).

Diese Sicht von Mathematik wird auch von ausgewiesenen Experten gestützt: So schreibt R. Thom (zitiert nach [Wi81], S. 130): „In good teaching one introduces new concepts, ideas etc. by using them, one explains their rules of interaction with primitive elements one has assumed to exist, one makes them familiar through handling these rules. It is only later that one will be able to give the abstract definition which allows one to verify the consistency of the theory extended in this way. Mathematics, even in its most elaborate form, has never proceeded otherwise (except perhaps for certain gratuitous generalizations of algebraic theories).”

2.2 Abgrenzung zum Historisch-Genetischen Ansatz

Oft wird bei „genetisch“ an „historisch-genetisch“ gedacht, wie es ursprünglich z.B. von Comenius propagiert wurde: „Am besten also, am leichtesten und am sichersten werden die Dinge so erkannt, wie sie entstanden sind [...] Daher möge der Gang der Lehre dem Gang der Tatsachen folgen und das Frühere zuerst, das Spätere nachher bringen“ ([Co61], S. 200). Ein solcher Ansatz möchte mit den Schülern das *tatsächliche* Werden eines Unterrichtsgegenstandes in der Geschichte nachvollziehen. Dies birgt eine Reihe von Schwierigkeiten. So ist es im Schulunterricht normalerweise wenig sinnvoll, jeden historischen Umweg bei der Lösungsfindung eines Problems nachzuvollziehen.

Außerdem vernachlässigt ein historisch-genetischer Lehrgang die eigene Ordnung, die den einzelnen Lerninhalten inne wohnt. Ein solcher Ansatz birgt auch die Gefahr in sich, die Vorerfahrungen der Schüler zu negieren, insbesondere dann, wenn, wie in der Informatik üblich, neue Ansätze und ihre Umsetzungen unter den Schülern schon weit verbreitet sind.

Genetisch ist hier also so zu verstehen, wie dem Zitat von Wittenberg weiter oben zu entnehmen ist, dass die Inhalte des Unterrichts so zu organisieren sind, wie sie ursprünglich hätten erschlossen werden können, orientiert an der Sachlogik des untersuchten Objekts und am Denken der Schüler.

2.3 Genetischer Informatikunterricht

Wie gezeigt wurde, hat das Konzept des genetischen Unterrichts in der Didaktik der Mathematik breite Unterstützung erfahren. In der aktuellen Literatur zur Didaktik der Informatik wird genetischer Unterricht aber bestenfalls erwähnt (z.B. bei [Hu06]). Dies ist bedauerlich, denn Informatikunterricht bietet in besonderer Weise die Chance, das Neue von Anfang an genetisch aufzubauen, weil man viele Gegenstände aktiv konstruieren kann: Man beginnt mit kleinen, einfachen Bausteinen und setzt nachher ein komplexes Produkt zusammen.

Weil man einen aktuellen, modernen Unterricht bieten möchte, ist man andererseits in der Informatik leicht der Versuchung ausgesetzt, neue Konzepte und Techniken den Schülern als Fertigprodukte vorzusetzen. Ein solcher Unterricht zielt darauf, das bereits strukturierte Produkt den Schülern zu vermitteln. Es liegt im Wesen der Informatik zu strukturieren und zu automatisieren und für die Lernenden ist eben dieser Prozess wichtig. Es ist für die Schüler nicht förderlich, gleich das strukturierte und automatisierte Endergebnis präsentiert zu bekommen.

2.3.1 Nicht genetische Ansätze im Informatikunterricht

Informatikunterricht steht immer in der Versuchung, das Neue für besonders gut und wichtig zu halten und deswegen den Schülern die letzten Modetrends zu präsentieren ohne den Weg dorthin mitzugehen. Als z.B. vor 20 Jahren Transputer (parallele Prozessoren) modern waren, wurde vorgeschlagen, diese im Informatikunterricht zu behandeln: Das ist nicht per se verkehrt, aber aus Mangel an Unterrichtszeit verführt ein solches Thema zu nicht genetischem Vorgehen.

Um es klar zu stellen: Genetischer Unterricht soll selbstverständlich aktuelle Themen der Informatik behandeln, insbesondere auch um die Motivation der Schüler aufrecht zu erhalten und um an ihre Vorerfahrungen im alltäglichen Umgang mit Computern anzuknüpfen. Man läuft dabei allerdings schnell Gefahr, diese Themen den Schülern als fertige Wunderwerke zu präsentieren, statt diese mit ihnen selbst zu entwickeln und so das Verständnis zu fördern. Ein Beispiel zur genetischen Behandlung eines aktuellen Themas aus der Informatik wird im nächsten Kapitel vorgestellt.

Auch der hessische Lehrplan Informatik kann ein nicht genetisches Vorgehen provozieren, wenn man sich die detaillierten Vorgaben betrachtet und diese Stück für Stück abarbeitet, statt die dahinter stehende Struktur zu analysieren und den Kurs genetisch zu strukturieren. Dies wird in Kapitel 3 genauer dargestellt.

2.3.2 Charakterisierung von genetischem Informatikunterricht

Für einen zeitgemäßen genetischen Informatikunterricht (GIU) soll folgende Charakterisierung als Arbeitsdefinition verwendet werden:

- (GIU1) GIU setzt bei den Vorerfahrungen der Schüler an.
- (GIU2) GIU fasst informatische Begriffe als „gewordene“ auf und will ihr mögliches „Werden“ im Lernprozess nachvollziehen lassen.²
- (GIU3) Entwicklung von einem einfachen Anfangspunkt aus. Vom Einfachen zum Komplexen, vom Anschaulichen zum Abstrakten.
- (GIU4) GIU stellt den Begriff der Entwicklung ins Zentrum der Informatik-Didaktik. Konzepte werden entwickelt, nicht präsentiert. Insbesondere muss erst eine Problemlage von den Schülerinnen und Schülern erkannt werden, und eine Problemlösung gewünscht werden, bevor diese erarbeitet werden kann.
- (GIU5) Formalismus sollte minimiert und dort, wo er unverzichtbar ist, entwickelt werden.
- (GIU6) Ein „Lernen auf Vorrat“ ist zu vermeiden.
- (GIU7) Es ist möglich und oft sinnvoll, bestimmte Themen aus dem deduktiven System herausgelöst zu behandeln, ohne vorher alle Grundlagen zu klären. Die Notwendigkeit der Behandlung dieser Grundlagen wird an der entsprechenden Stelle sofort klar und ist dann immer noch und sogar leichter möglich³.
- (GIU8) Anwendungen haben eine hohe Bedeutung. Die Schüler sollen Konzepte nicht nur lernen, sondern von Anfang an praktisch umsetzen.

2.3.3 Informatik im Kontext (IniK)

Genetischer Informatikunterricht (GIU) und Informatik im Kontext (IniK) haben einige Gemeinsamkeiten (Anknüpfen an das Vorwissen der Schüler, weg von der Fachsystematik). Außerdem ist IniK eine gute Unterstützung des genetischen Informatikunterrichts, da sie viele Beispiele liefert, die gut genetisch aufgezogen werden können und ähnliche Ziele verfolgt.

² Dadurch bekommen die prozessbezogenen Kompetenzen im Sinne der Bildungsstandards der Gesellschaft für Informatik besondere Bedeutung.

³ So eröffnet sich u.U. eine Möglichkeit zur Binnendifferenzierung.

Genetischer Informatikunterricht bietet mehr als Informatik im Kontext. So geht GIU von einfachen Ausgangspunkten aus und entwickelt daraus komplexe Konzepte, während IniK existierende komplexe Konzepte in ihrem Kontext betrachtet. So lassen sich im genetischen Informatikunterricht auch theoretische Konzepte erschließen, die nicht in das Konzept der IniK passen. Als Beispiel lässt sich das Halteproblem anführen, das in einem genetischen Informatikunterricht über das Debugging beim Programmieren erarbeitet werden könnte.

3 Planung einer Unterrichtseinheit nach dem genetischen Prinzip

Im Rahmen eines Schulversuchs zum genetischen Informatikunterricht wurde in einer Unterrichtseinheit das Programmieren anhand von CGI-Skripten eingeführt. Im Folgenden werden die Planung und die Durchführung dieser Unterrichtsreihe beschrieben. Daran soll gezeigt werden, welche Rolle das genetische Prinzip spielen kann und welchen Vorteil ein genetisches Vorgehen hat.

3.1 Der Schulversuch

Die Lage des Informatikunterrichts gibt Anlass zur Sorge. Viele Lehrer aus verschiedenen Bundesländern berichten, dass die Anwahl der Kurse insbesondere in der Qualifizierungsphase hinter den Erwartungen zurückbleibt. Das gilt sogar an solchen Schulen, die die Informatik in der Sekundarstufe I besonders fördern, z.B. im Rahmen des Wahlpflichtunterrichts. Informatik erscheint also offensichtlich als ein schwieriges Fach, das den Impuls des verbreiteten Interesses an Computern nicht in ausreichendem Maße in Interesse an den Grundlagen der Informatik umsetzen kann. Es scheint, dass gerade der Zeitpunkt der Transformation von der informellen Informatik mit eher spielerisch-experimentellem Ansatz (wie vielerorts in der Sekundarstufe I praktiziert) zur Wissenschaftspropädeutik kritisch ist, dass also die wissenschaftlichen Konzepte der Informatik nicht auf fruchtbaren Boden fallen.

Ein wichtiger Punkt dürfte auch der hohe Aufwand für technische Aspekte des Programmierens sein. Traditionelle Programmiersprachen wie Delphi und Java fordern eine umfangreiche Einarbeitung und überfrachten den Lernenden mit einer Vielzahl von Details. Dies bringt die Gefahr mit sich, dass viele der von Schülern als attraktiv empfundenen Themen (siehe z.B. [RO09]) zu kurz kommen. Der Versuch soll erproben, wie sich die Ideen des genetischen Unterrichts im zeitgemäßen Informatikunterricht umsetzen lassen. Zeitgemäß bedeutet dabei insbesondere, dass die Vorerfahrungen der Schüler und Schülerinnen mit Computeranwendungen, Internet (insbesondere auch dem Web 2.0) und Mobilfunktechniken genutzt werden.

Im Laufe des Versuchs soll Unterrichtsmaterial entwickelt werden, das den genetischen Zugang umsetzt. Dieses Material wird die Chancen des interaktiven und durch Literale unterstützten Arbeitens mit Python intensiv nutzen. Gleichzeitig soll erforscht werden, ob eine – ggf. teilweise – Übertragung des Konzeptes auf weiter verbreitete Programmiersprachen möglich ist.

3.2 Vorerfahrungen der Schüler bzgl. interaktiver Webseiten

Die Schüler haben sich in ihrem bisherigen Informatikunterricht in diesem Kurs mit dem Internet beschäftigt und gelernt, Internetseiten in XHTML und CSS zu erstellen und dabei Textdateien als ein Mittel zur Beschreibung statischer Inhalte (Struktur und Formatierung) kennengelernt.

Internetseiten mit rein statischen Inhalten sind in der Erfahrungswelt der Schüler in Zeiten des Web 2.0 eher selten geworden. Die unter den Schülern verbreiteten Internetseiten (Facebook, YouTube, Google, ...) sind interaktiv, können also auf Benutzereingaben reagieren. Dazu muss der Computer eingegebene Daten verarbeiten und der Entwickler nicht mehr nur statischen Inhalt festlegen, sondern Handlungsanweisungen für den Computer in Form von Programmen definieren (programmieren).

Um also an die Vorerfahrungen der Schüler anzuknüpfen, soll das Programmieren anhand von CGI-Skripten eingeführt werden, die Daten aus XHTML-Formularen entgegen nehmen, verarbeiten und eine XHTML-Ausgabe erzeugen⁴.

An diesem Vorgehen sieht man auch, dass genetisch nicht unbedingt bedeuten muss, die tatsächliche historische Entstehung nachzuvollziehen, sondern auch bedeuten kann, einen möglichen Weg nachzugehen, wie der zu lernende Begriff / das zu lernende Objekt (historisch) hätte entstehen können. Das Programmieren im ursprünglichen Sinne wurde historisch gesehen natürlich nicht als Erweiterung der Funktionalitäten von Webseiten „erfunden“!

Es muss beachtet werden, dass natürlich auch bei diesem Thema nicht automatisch sichergestellt ist, dass man genetisch vorgeht. So besteht bspw. die Gefahr, den Schülern wichtige Konzepte wie das Client-Server-Prinzip als fertige Gegebenheit zu präsentieren. Daher wurde in der Phase der Einführung des Themas CGI darauf Wert gelegt, dass die Schüler von sich aus erkennen, dass die Ausführung des Skriptes auf dem Server geschehen muss, nachdem sie das Client-Server-Prinzip bereits zu Beginn des Kurshalbjahres im Rahmen eines Experiments ausgiebig erfahren haben⁵. Dies wird insbesondere auch dadurch deutlich und für die Schüler sichtbar, dass die HTML- und die Python-Datei auf den Server hochgeladen werden müssen und die Dateizugriffsrechte entsprechend gesetzt werden müssen.

3.3 Einordnung in den hessischen Lehrplan

Der hessische Lehrplan sieht für das erste Jahr der gymnasialen Oberstufe das Thema Internet und die Einführung in das Programmieren vor. So lernen die Schüler den Aufbau und die Funktionsweise des Internets inklusive des Client-Server-Prinzips kennen und erstellen eigene Internetseiten. Der Lehrplan schreibt zum Thema HTML vor, dass die Schüler auch HTML-Formulare erstellen sollen.

⁴ Eine detaillierte inhaltliche Darstellung der CGI-Programmierung ist im Rahmen dieses Artikels leider nicht möglich. Deshalb sei hier auf entsprechende Literatur (z.B. [We06], S. 529f) verwiesen.

⁵ Dabei haben die Schüler mit ihren Computern schrittweise mithilfe von Netzwerkkabeln und Switches ein Netzwerk aufgebaut und immer mehr Dienste installiert um die Funktionalitäten des Internets abzubilden.

Dies ist ein Thema mit hohem informatischen Bildungswert, allerdings nur, wenn man mit den Formularen auch „etwas anfangen kann“, sprich diese auswertet. Das einfache Verschicken von Emails über die mailto-Funktion ist unbefriedigend und meist aufgrund technischer Schwierigkeiten nicht umsetzbar⁶. Außerdem findet die Auswertung des Formulars dabei durch den Computer im Verborgenen statt, ohne dass der Schüler dies nachvollziehen kann. Es scheint also sinnvoll, gerade hier mit dem Programmieren zu beginnen und zwar anhand einer automatisierten Verarbeitung der Daten aus Formularen. Diese kann clientseitig erfolgen (mittels JavaScript) oder serverseitig über CGI-Skripte, die in verschiedenen Programmiersprachen möglich sind, meist wird allerdings PHP verwendet.

JavaScript und PHP eignen sich durchaus für eine Einführung in das Programmieren, wie zahlreiche ausgearbeitete Unterrichtskonzepte in beiden Sprachen nahelegen. Allerdings sind beide Programmiersprachen auf ihren eigenen sehr eng gefassten Einsatzbereich beschränkt. Außerdem ist die clientseitige Verarbeitung der Daten von geringerer Bedeutung, da die CGI-Programmierung später im Zusammenhang mit dem Zugriff auf Datenbanken zusätzlich an Bedeutung gewinnt und dann wieder aufgegriffen werden kann. Es bietet sich daher an, CGI-Skripte zu programmieren und dazu eine Programmiersprache zu verwenden, die den ganzen Kurs hindurch bis zum Abitur eingesetzt werden kann. PHP ist im CGI-Bereich der Quasi-Standard, lässt sich darüber hinaus allerdings wenig einsetzen. Python bietet beide Möglichkeiten. Es lassen sich leicht CGI-Skripte schreiben (wie dieser Beitrag zeigen will), und auch für den weiteren Oberstufenunterricht ist Python einsetzbar.

Dass Python sich für die Einführung in das Programmieren (insbesondere an der Schule) eignet, zeigt z.B. Ingo Linkweiler in seiner Diplomarbeit [Li02]. Auch das MIT ist seit einiger Zeit in den einführenden Programmierkursen auf Python umgestiegen.

3.4 Inhaltliche Planung der Unterrichtseinheit

Die Unterrichtseinheit baut auf einem vorhergehenden kleinen Unterrichtsprojekt auf, in dem die Schüler eine mehrere Seiten umfassende Website zu einem Reiseziel erstellten, nachdem sie die Grundlagen von XHTML erlernt hatten. Nach der abschließenden Präsentation der Ergebnisse folgte eine Reflexionsrunde, in der die Schüler überlegen sollten, wie sie ihr Projekt gerne noch erweitern würden und dabei unterscheiden, ob sie es noch nicht umgesetzt haben, weil sie selbst es noch lernen müssten oder weil XHTML diese Möglichkeiten generell nicht bietet. Aus dieser Diskussion ergaben sich verschiedene Gesichtspunkte, von denen einer immer wieder genannt wurde: Interaktivität. Hieraus entsteht die nötige Einsicht und Motivation, etwas Neues lernen zu wollen.

⁶ Dazu wäre ein installierter und konfigurierter Email-Client auf dem Computer notwendig, was Benutzerkonten auf den Schulcomputern voraussetzt. Dies ist in den meisten Schulen nicht gegeben.

Der Lehrer führt den Schülern eine Webseite vor, die ein Formular enthält, das Daten des Benutzers (Name, Vorname, Alter, Gewicht) abfragt. Nach Absenden des Formulars erhält der Benutzer eine XHTML-Seite, die die eingegebenen Daten aufbereitet (z.B. das Alter in Hundejahre umrechnet und das Gewicht auf dem Mond angibt)⁷. Die Schüler analysierten anschließend die Webseite. Indem sie die Quelltexte betrachteten, konnten sie sehen, dass das Formular mit einer Datei mit der Endung .py verknüpft (die offensichtlich auf dem Server liegt) und dass mit dieser dann durch den Server eine Ausgabe in Form einer HTML-Seite erzeugt wird. Daraus ergibt sich die Frage, wie der Server Texte verarbeiten kann.

Um diese Frage zu beantworten, bekamen die Schüler den Quelltext der XHTML-Seite und des Python-Skripts ausgeteilt und konnten dieses ohne größere Schwierigkeiten sich selbst und den anderen erklären. Die im Quelltext ausgeteilten Dateien wurden den Schülern auch elektronisch zur Verfügung gestellt. Sie ergänzten das Skript um einen weiteren Absatz und hatten so bereits ihre erste Programmiererfahrung mit Python gesammelt.

Das Lernen durch Selbsterklären von ausgearbeiteten Lösungsbeispielen (wie hier) ist u.a. von Renkl ([Re08] Kapitel 4.3.3, S.123ff) und Zöttl ([Zö10]) untersucht worden. Wie Renkl zeigt, ist dieses Konzept vor allem dann erfolgreich, wenn viele Selbsterklärungsaktivitäten vom Schüler gefordert werden. Dies lässt sich in der Informatik besonders leicht umsetzen. In dem hier dargestellten Fall müssen die Schüler sich anhand des Quelltextes z.B. erklären, wie eine Zeichenkette mit einer Zahl multipliziert werden kann. Weitere Selbsterklärungsaktivität wird hervorgerufen, wenn die Schüler das Skript erweitern sollen.

Auf dem bisherigen Unterrichtsverlauf aufbauend können nun alle wesentlichen Themen eines Kurses zur Einführung in das Programmieren anhand der CGI-Skripte behandelt werden, um später darauf aufzubauen. So werden Verzweigungen genutzt, um abhängig von den vom Benutzer eingegebenen Daten spezifische Rückgaben zu machen oder auch Checkboxen abzufragen. Jetzt lässt sich ein Feedbackformular erstellen, das als Erweiterung für das Reiseziel-Projekt eingeführt wird. Außerdem könnte sich dann die Frage stellen, ob es nicht möglich ist, einen Emailverteiler zu verwalten. Die abgesendeten Daten lassen sich leicht in eine Textdatei speichern und dann mit Hilfe einer Schleife zeilenweise auslesen. Um andere Schleifentypen und verschachtelte Schleifen einzusetzen, ließe sich z.B. ein Terminkalender erstellen, der wiederum eine ganze Menge algorithmischer Fragestellungen erlaubt.

⁷ Das Beispiel ist dem Buch „Python programming for the absolute beginner“ ([Da06], S. 17 ff) entnommen und für die CGI-Programmierung adaptiert.

4 Fazit

Der vorliegende Artikel zeigt, dass das Genetische Prinzip die Informatikdidaktik entscheidend bereichern kann, weil ein genetischer Unterricht enger am Schüler orientiert ist als Unterricht, der einem streng fachsystematischen Konzept folgt und damit schlussendlich erfolgreicher sein kann. Außerdem bietet der genetische Unterricht viele Anknüpfungspunkte an wichtige didaktisch/methodische Konzepte (Kompetenzorientierung, Binnendifferenzierung,...). Informatik ist besonders geeignet, genetisch unterrichtet zu werden, weil das Strukturieren und Automatisieren wesentliche Grundsätze der Informatik sind und es daher unabdingbar erscheint, die Schüler diese Prozesse beim Erlernen neuen Stoffes mit erleben zu lassen.

Dies wurde am Beispiel der CGI-Programmierung gezeigt, die ein aktuelles und wichtiges Thema der Informatik ist und aus didaktischer Perspektive eine sehr gute Möglichkeit bietet, eine Brücke zu schlagen zwischen dem Medium Text zur Beschreibung von Inhalt zum Medium Text zur Beschreibung von Abläufen; konkret gesprochen vom Gestalten von statischen Webseiten in (X)HTML zum Programmieren, in diesem Fall von CGI-Skripten in Python.

Im Rahmen des Schulversuchs sollen weitere Beispiele für genetischen Informatikunterricht gesammelt werden und auch damit weitere Kriterien für genetischen Informatikunterricht erstellt werden um das Konzept des genetischen Informatikunterrichts weiter auszubauen und weiter zu präzisieren. Über den gesamten Verlauf des Schulversuchs während der drei Jahre an zwei Schulen soll beobachtet werden, wie sich die konsequente genetische Planung z.B. auf die Abwahltendenz, die Leistungen der Schüler, die Motivation, die Selbstwirksamkeitserwartung im Vergleich zu Parallelkursen verändert.

Literaturverzeichnis

- [Co61] Comenius, J. A.: Grosse Didaktik. VEB Verl. Volk u. Wissen, 1961.
- [Da06] Dawson, M.: Python programming for the absolute beginner. Thomson Course Technology, Boston Mass., 2006.
- [Hu06] Humbert, L.: Didaktik der Informatik. Vieweg + Teubner, Wiesbaden 2006.
- [Kl24] Klein, F.: Elementarmathematik vom höheren Standpunkte aus, Bd. 1. Berlin-Göttingen-Heidelberg, 1924.
- [Li02] Linkweiler, I.: Eignet sich die Skriptsprache Python für schnelle Entwicklungen im Softwareentwicklungsprozess? <http://www.ingo-linkweiler.de/diplom/Diplomarbeit.pdf>.
- [Re08] Renkl, A.: Lehrbuch Pädagogische Psychologie, 1.Aufl., Huber, Bern 2008.
- [RO09] Rabel, M.; Oldenburg, R.: Konzepte, Modelle und Projekte im Informatikunterricht - Bewertungen und Erwartungen von Studenten. In (Koerber, B. Hrsg.): Zukunft braucht Herkunft. 25 Jahre INFOS ; INFOS 2009. Ges. für Informatik, Bonn, 2009; S. 146–156.
- [We06] Weigend, M.: Objektorientierte Programmierung mit Python. mitp, Bonn, 2006.
- [Wi63] Wittenberg, A. I.: Bildung und Mathematik. 2. Aufl., Klett, Stuttgart 1990.
- [Wi81] Wittmann, E. Ch: Grundfragen des Mathematikunterrichts. 6. Aufl., Vieweg, Braunschweig 1981.
- [Zö10] Zöttl, L.: Modellierungskompetenz fördern mit heuristischen Lösungsbeispielen. Univ., Diss.-München, 2009. Franzbecker, Hildesheim, 2010.

GI-Edition Lecture Notes in Informatics

- P-1 Gregor Engels, Andreas Oberweis, Albert Zündorf (Hrsg.): Modellierung 2001.
- P-2 Mikhail Godlevsky, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications, ISTA'2001.
- P-3 Ana M. Moreno, Reind P. van de Riet (Hrsg.): Applications of Natural Language to Information Systems, NLDB'2001.
- P-4 H. Wörn, J. Mühling, C. Vahl, H.-P. Meinzer (Hrsg.): Rechner- und sensorgestützte Chirurgie; Workshop des SFB 414.
- P-5 Andy Schürr (Hg.): OMER – Object-Oriented Modeling of Embedded Real-Time Systems.
- P-6 Hans-Jürgen Appelrath, Rolf Beyer, Uwe Marquardt, Heinrich C. Mayr, Claudia Steinberger (Hrsg.): Unternehmen Hochschule, UH'2001.
- P-7 Andy Evans, Robert France, Ana Moreira, Bernhard Rumpe (Hrsg.): Practical UML-Based Rigorous Development Methods – Countering or Integrating the extremists, pUML'2001.
- P-8 Reinhard Keil-Slawik, Johannes Magenheimer (Hrsg.): Informatikunterricht und Medienbildung, INFOS'2001.
- P-9 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Innovative Anwendungen in Kommunikationsnetzen, 15. DFN Arbeitstagung.
- P-10 Mirjam Minor, Steffen Staab (Hrsg.): 1st German Workshop on Experience Management: Sharing Experiences about the Sharing Experience.
- P-11 Michael Weber, Frank Kargl (Hrsg.): Mobile Ad-Hoc Netzwerke, WMAN 2002.
- P-12 Martin Glinz, Günther Müller-Luschnat (Hrsg.): Modellierung 2002.
- P-13 Jan von Knop, Peter Schirmbacher and Viljan Mahni_ (Hrsg.): The Changing Universities – The Role of Technology.
- P-14 Robert Tolksdorf, Rainer Eckstein (Hrsg.): XML-Technologien für das Semantic Web – XSW 2002.
- P-15 Hans-Bernd Bludau, Andreas Koop (Hrsg.): Mobile Computing in Medicine.
- P-16 J. Felix Hampe, Gerhard Schwabe (Hrsg.): Mobile and Collaborative Business 2002.
- P-17 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Zukunft der Netze –Die Verletzbarkeit meistern, 16. DFN Arbeitstagung.
- P-18 Elmar J. Sinz, Markus Plaha (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2002.
- P-19 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund.
- P-20 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund (Ergänzungsband).
- P-21 Jörg Desel, Mathias Weske (Hrsg.): Promise 2002: Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen.
- P-22 Sigrid Schubert, Johannes Magenheimer, Peter Hubwieser, Torsten Brinda (Hrsg.): Forschungsbeiträge zur “Didaktik der Informatik” – Theorie, Praxis, Evaluation.
- P-23 Thorsten Spitta, Jens Borchers, Harry M. Sneed (Hrsg.): Software Management 2002 – Fortschritt durch Beständigkeit
- P-24 Rainer Eckstein, Robert Tolksdorf (Hrsg.): XMIDX 2003 – XML-Technologien für Middleware – Middleware für XML-Anwendungen
- P-25 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Commerce – Anwendungen und Perspektiven – 3. Workshop Mobile Commerce, Universität Augsburg, 04.02.2003
- P-26 Gerhard Weikum, Harald Schöning, Erhard Rahm (Hrsg.): BTW 2003: Datenbanksysteme für Business, Technologie und Web
- P-27 Michael Kroll, Hans-Gerd Lipinski, Kay Melzer (Hrsg.): Mobiles Computing in der Medizin
- P-28 Ulrich Reimer, Andreas Abecker, Steffen Staab, Gerd Stumme (Hrsg.): WM 2003: Professionelles Wissensmanagement – Erfahrungen und Visionen
- P-29 Antje Düsterhöft, Bernhard Thalheim (Eds.): NLDB'2003: Natural Language Processing and Information Systems
- P-30 Mikhail Godlevsky, Stephen Liddle, Heinrich C. Mayr (Eds.): Information Systems Technology and its Applications
- P-31 Arslan Brömmel, Christoph Busch (Eds.): BIOSIG 2003: Biometrics and Electronic Signatures

- P-32 Peter Hubwieser (Hrsg.): Informatische Fachkonzepte im Unterricht – INFOS 2003
- P-33 Andreas Geyer-Schulz, Alfred Taudes (Hrsg.): Informationswirtschaft: Ein Sektor mit Zukunft
- P-34 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 1)
- P-35 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 2)
- P-36 Rüdiger Grimm, Hubert B. Keller, Kai Rannenberg (Hrsg.): Informatik 2003 – Mit Sicherheit Informatik
- P-37 Arndt Bode, Jörg Desel, Sabine Rathmayer, Martin Wessner (Hrsg.): DeLFI 2003: e-Learning Fachtagung Informatik
- P-38 E.J. Sinz, M. Plaha, P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2003
- P-39 Jens Nedon, Sandra Frings, Oliver Göbel (Hrsg.): IT-Incident Management & IT-Forensics – IMF 2003
- P-40 Michael Rebstock (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2004
- P-41 Uwe Brinkschulte, Jürgen Becker, Dietmar Fey, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle, Thomas Runkler (Edts.): ARCS 2004 – Organic and Pervasive Computing
- P-42 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Economy – Transaktionen und Prozesse, Anwendungen und Dienste
- P-43 Birgitta König-Ries, Michael Klein, Philipp Obreiter (Hrsg.): Persistence, Scalability, Transactions – Database Mechanisms for Mobile Applications
- P-44 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): Security, E-Learning, E-Services
- P-45 Bernhard Rumpe, Wolfgang Hesse (Hrsg.): Modellierung 2004
- P-46 Ulrich Flegel, Michael Meier (Hrsg.): Detection of Intrusions of Malware & Vulnerability Assessment
- P-47 Alexander Prosser, Robert Krimmer (Hrsg.): Electronic Voting in Europe – Technology, Law, Politics and Society
- P-48 Anatoly Doroshenko, Terry Halpin, Stephen W. Liddle, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications
- P-49 G. Schiefer, P. Wagner, M. Morgenstern, U. Rickert (Hrsg.): Integration und Datensicherheit – Anforderungen, Konflikte und Perspektiven
- P-50 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 1) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm
- P-51 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 2) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm
- P-52 Gregor Engels, Silke Seehusen (Hrsg.): DELFI 2004 – Tagungsband der 2. e-Learning Fachtagung Informatik
- P-53 Robert Giegerich, Jens Stoye (Hrsg.): German Conference on Bioinformatics – GCB 2004
- P-54 Jens Borchers, Ralf Kneuper (Hrsg.): Softwaremanagement 2004 – Outsourcing und Integration
- P-55 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): E-Science und Grid Ad-hoc-Netze Medienintegration
- P-56 Fernand Feltz, Andreas Oberweis, Benoit Otjacques (Hrsg.): EMISA 2004 – Informationssysteme im E-Business und E-Government
- P-57 Klaus Turowski (Hrsg.): Architekturen, Komponenten, Anwendungen
- P-58 Sami Beydeda, Volker Gruhn, Johannes Mayer, Ralf Reussner, Franz Schweiggert (Hrsg.): Testing of Component-Based Systems and Software Quality
- P-59 J. Felix Hampe, Franz Lehner, Key Pousttchi, Kai Ranneberg, Klaus Turowski (Hrsg.): Mobile Business – Processes, Platforms, Payments
- P-60 Steffen Friedrich (Hrsg.): Unterrichtskonzepte für informatische Bildung
- P-61 Paul Müller, Reinhard Gotzhein, Jens B. Schmitt (Hrsg.): Kommunikation in verteilten Systemen
- P-62 Federrath, Hannes (Hrsg.): „Sicherheit 2005“ – Sicherheit – Schutz und Zuverlässigkeit
- P-63 Roland Kaschek, Heinrich C. Mayr, Stephen Liddle (Hrsg.): Information Systems – Technology and its Applications

- P-64 Peter Liggesmeyer, Klaus Pohl, Michael Goedicke (Hrsg.): Software Engineering 2005
- P-65 Gottfried Vossen, Frank Leymann, Peter Lockemann, Wolffried Stucky (Hrsg.): Datenbanksysteme in Business, Technologie und Web
- P-66 Jörg M. Haake, Ulrike Lucke, Djamshid Tavangarian (Hrsg.): DeLFI 2005: 3. deutsche e-Learning Fachtagung Informatik
- P-67 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 1)
- P-68 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 2)
- P-69 Robert Hirschfeld, Ryszard Kowalczyk, Andreas Polze, Matthias Weske (Hrsg.): NODE 2005, GSEM 2005
- P-70 Klaus Turowski, Johannes-Maria Zaha (Hrsg.): Component-oriented Enterprise Application (COAE 2005)
- P-71 Andrew Torda, Stefan Kurz, Matthias Rarey (Hrsg.): German Conference on Bioinformatics 2005
- P-72 Klaus P. Jantke, Klaus-Peter Fährnrich, Wolfgang S. Wittig (Hrsg.): Marktplatz Internet: Von e-Learning bis e-Payment
- P-73 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): "Heute schon das Morgen sehen"
- P-74 Christopher Wolf, Stefan Lucks, Po-Wah Yau (Hrsg.): WEWoRC 2005 – Western European Workshop on Research in Cryptology
- P-75 Jörg Desel, Ulrich Frank (Hrsg.): Enterprise Modelling and Information Systems Architecture
- P-76 Thomas Kirste, Birgitta König-Riess, Key Poustchi, Klaus Turowski (Hrsg.): Mobile Informationssysteme – Potentiale, Hindernisse, Einsatz
- P-77 Jana Dittmann (Hrsg.): SICHERHEIT 2006
- P-78 K.-O. Wenkel, P. Wagner, M. Morgens-tern, K. Luzi, P. Eisermann (Hrsg.): Land- und Ernährungswirtschaft im Wandel
- P-79 Bettina Biel, Matthias Book, Volker Gruhn (Hrsg.): Softwareengineering 2006
- P-80 Mareike Schoop, Christian Huemer, Michael Rebstock, Martin Bichler (Hrsg.): Service-Oriented Electronic Commerce
- P-81 Wolfgang Karl, Jürgen Becker, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle (Hrsg.): ARCS'06
- P-82 Heinrich C. Mayr, Ruth Breu (Hrsg.): Modellierung 2006
- P-83 Daniel Huson, Oliver Kohlbacher, Andrei Lupas, Kay Nieselt and Andreas Zell (eds.): German Conference on Bioinformatics
- P-84 Dimitris Karagiannis, Heinrich C. Mayr, (Hrsg.): Information Systems Technology and its Applications
- P-85 Witold Abramowicz, Heinrich C. Mayr, (Hrsg.): Business Information Systems
- P-86 Robert Krimmer (Ed.): Electronic Voting 2006
- P-87 Max Mühlhäuser, Guido Röbling, Ralf Steinmetz (Hrsg.): DELFI 2006: 4. e-Learning Fachtagung Informatik
- P-88 Robert Hirschfeld, Andreas Polze, Ryszard Kowalczyk (Hrsg.): NODE 2006, GSEM 2006
- P-90 Joachim Schelp, Robert Winter, Ulrich Frank, Bodo Rieger, Klaus Turowski (Hrsg.): Integration, Informationslogistik und Architektur
- P-91 Henrik Stormer, Andreas Meier, Michael Schumacher (Eds.): European Conference on eHealth 2006
- P-92 Fernand Feltz, Benoît Otjacques, Andreas Oberweis, Nicolas Poussing (Eds.): AIM 2006
- P-93 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 1
- P-94 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 2
- P-95 Matthias Weske, Markus Nüttgens (Eds.): EMISA 2005: Methoden, Konzepte und Technologien für die Entwicklung von dienstbasierten Informationssystemen
- P-96 Saartje Brockmans, Jürgen Jung, York Sure (Eds.): Meta-Modelling and Ontologies
- P-97 Oliver Göbel, Dirk Schadt, Sandra Frings, Hardo Hase, Detlef Günther, Jens Nedon (Eds.): IT-Incident Mangament & IT-Forensics – IMF 2006

- P-98 Hans Brandt-Pook, Werner Simonsmeier und Thorsten Spitta (Hrsg.): Beratung in der Softwareentwicklung – Modelle, Methoden, Best Practices
- P-99 Andreas Schwill, Carsten Schulte, Marco Thomas (Hrsg.): Didaktik der Informatik
- P-100 Peter Forbrig, Günter Siegel, Markus Schneider (Hrsg.): HDI 2006: Hochschuldidaktik der Informatik
- P-101 Stefan Böttinger, Ludwig Theuvsen, Susanne Rank, Marlies Morgenstern (Hrsg.): Agrarinformatik im Spannungsfeld zwischen Regionalisierung und globalen Wertschöpfungsketten
- P-102 Otto Spaniol (Eds.): Mobile Services and Personalized Environments
- P-103 Alfons Kemper, Harald Schöning, Thomas Rose, Matthias Jarke, Thomas Seidl, Christoph Quix, Christoph Brochhaus (Hrsg.): Datenbanksysteme in Business, Technologie und Web (BTW 2007)
- P-104 Birgitta König-Ries, Franz Lehner, Rainer Malaka, Can Türker (Hrsg.): MMS 2007: Mobilität und mobile Informationssysteme
- P-105 Wolf-Gideon Bleek, Jörg Raasch, Heinz Züllighoven (Hrsg.): Software Engineering 2007
- P-106 Wolf-Gideon Bleek, Henning Schwentner, Heinz Züllighoven (Hrsg.): Software Engineering 2007 – Beiträge zu den Workshops
- P-107 Heinrich C. Mayr, Dimitris Karagiannis (eds.): Information Systems Technology and its Applications
- P-108 Arslan Brömmе, Christoph Busch, Detlef Hühnlein (eds.): BIOSIG 2007: Biometrics and Electronic Signatures
- P-109 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.): INFORMATIK 2007 Informatik trifft Logistik Band 1
- P-110 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.): INFORMATIK 2007 Informatik trifft Logistik Band 2
- P-111 Christian Eibl, Johannes Magenheimer, Sigrid Schubert, Martin Wessner (Hrsg.): DeLFI 2007: 5. e-Learning Fachtagung Informatik
- P-112 Sigrid Schubert (Hrsg.): Didaktik der Informatik in Theorie und Praxis
- P-113 Sören Auer, Christian Bizer, Claudia Müller, Anna V. Zhdanova (Eds.): The Social Semantic Web 2007 Proceedings of the 1st Conference on Social Semantic Web (CSSW)
- P-114 Sandra Frings, Oliver Göbel, Detlef Günther, Hardo G. Hase, Jens Nedon, Dirk Schadt, Arslan Brömmе (Eds.): IMF2007 IT-incident management & IT-forensics Proceedings of the 3rd International Conference on IT-Incident Management & IT-Forensics
- P-115 Claudia Falter, Alexander Schliep, Joachim Selbig, Martin Vingron and Dirk Walther (Eds.): German conference on bioinformatics GCB 2007
- P-116 Witold Abramowicz, Leszek Maciszek (Eds.): Business Process and Services Computing 1st International Working Conference on Business Process and Services Computing BPSC 2007
- P-117 Ryszard Kowalczyk (Ed.): Grid service engineering and management The 4th International Conference on Grid Service Engineering and Management GSEM 2007
- P-118 Andreas Hein, Wilfried Thoben, Hans-Jürgen Appelrath, Peter Jensch (Eds.): European Conference on ehealth 2007
- P-119 Manfred Reichert, Stefan Strecker, Klaus Turowski (Eds.): Enterprise Modelling and Information Systems Architectures Concepts and Applications
- P-120 Adam Pawlak, Kurt Sandkuhl, Wojciech Cholewa, Leandro Soares Indrusiak (Eds.): Coordination of Collaborative Engineering - State of the Art and Future Challenges
- P-121 Korbinian Herrmann, Bernd Bruegge (Hrsg.): Software Engineering 2008 Fachtagung des GI-Fachbereichs Softwaretechnik
- P-122 Walid Maalej, Bernd Bruegge (Hrsg.): Software Engineering 2008 - Workshopband Fachtagung des GI-Fachbereichs Softwaretechnik

- P-123 Michael H. Breitner, Martin Breunig, Elgar Fleisch, Ley Pousttchi, Klaus Turowski (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Technologien, Prozesse, Marktfähigkeit
Proceedings zur 3. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2008)
- P-124 Wolfgang E. Nagel, Rolf Hoffmann, Andreas Koch (Eds.)
9th Workshop on Parallel Systems and Algorithms (PASA)
Workshop of the GI/ITG Special Interest Groups PARS and PARVA
- P-125 Rolf A.E. Müller, Hans-H. Sundermeier, Ludwig Theuvsen, Stephanie Schütze, Marlies Morgenstern (Hrsg.)
Unternehmens-IT:
Führungsinstrument oder Verwaltungsbürde
Referate der 28. GIL Jahrestagung
- P-126 Rainer Gimnich, Uwe Kaiser, Jochen Quante, Andreas Winter (Hrsg.)
10th Workshop Software Reengineering (WSR 2008)
- P-127 Thomas Kühne, Wolfgang Reisig, Friedrich Steimann (Hrsg.)
Modellierung 2008
- P-128 Ammar Alkassar, Jörg Siekmann (Hrsg.)
Sicherheit 2008
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)
2.-4. April 2008
Saarbrücken, Germany
- P-129 Wolfgang Hesse, Andreas Oberweis (Eds.)
Sigsand-Europe 2008
Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems
- P-130 Paul Müller, Bernhard Neumair, Gabi Dreö Rodosek (Hrsg.)
1. DFN-Forum Kommunikationstechnologien Beiträge der Fachtagung
- P-131 Robert Krimmer, Rüdiger Grimm (Eds.)
3rd International Conference on Electronic Voting 2008
Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting.CC
- P-132 Silke Seehusen, Ulrike Lucke, Stefan Fischer (Hrsg.)
DeLFI 2008:
Die 6. e-Learning Fachtagung Informatik
- P-133 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik Band 1
- P-134 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik Band 2
- P-135 Torsten Brinda, Michael Fothe, Peter Hubwieser, Kirsten Schlüter (Hrsg.)
Didaktik der Informatik – Aktuelle Forschungsergebnisse
- P-136 Andreas Beyer, Michael Schroeder (Eds.)
German Conference on Bioinformatics GCB 2008
- P-137 Arslan Brömme, Christoph Busch, Detlef Hühnlein (Eds.)
BIOSIG 2008: Biometrics and Electronic Signatures
- P-138 Barbara Dinter, Robert Winter, Peter Chamoni, Norbert Gronau, Klaus Turowski (Hrsg.)
Synergien durch Integration und Informationslogistik
Proceedings zur DW2008
- P-139 Georg Herzwurm, Martin Mikusz (Hrsg.)
Industrialisierung des Software-Managements
Fachtagung des GI-Fachausschusses Management der Anwendungsentwicklung und -wartung im Fachbereich Wirtschaftsinformatik
- P-140 Oliver Göbel, Sandra Frings, Detlef Günther, Jens Nedon, Dirk Schadt (Eds.)
IMF 2008 - IT Incident Management & IT Forensics
- P-141 Peter Loos, Markus Nüttgens, Klaus Turowski, Dirk Werth (Hrsg.)
Modellierung betrieblicher Informationssysteme (MobIS 2008)
Modellierung zwischen SOA und Compliance Management
- P-142 R. Bill, P. Korduan, L. Theuvsen, M. Morgenstern (Hrsg.)
Anforderungen an die Agrarinformatik durch Globalisierung und Klimaveränderung
- P-143 Peter Liggesmeyer, Gregor Engels, Jürgen Münch, Jörg Dörr, Norman Riegel (Hrsg.)
Software Engineering 2009
Fachtagung des GI-Fachbereichs Softwaretechnik

- P-144 Johann-Christoph Freytag, Thomas Ruf, Wolfgang Lehner, Gottfried Vossen (Hrsg.)
Datenbanksysteme in Business, Technologie und Web (BTW)
- P-145 Knut Hinkelmann, Holger Wache (Eds.)
WM2009: 5th Conference on Professional Knowledge Management
- P-146 Markus Bick, Martin Breunig, Hagen Höpfner (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Entwicklung, Implementierung und Anwendung
4. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2009)
- P-147 Witold Abramowicz, Leszek Maciaszek, Ryszard Kowalczyk, Andreas Speck (Eds.)
Business Process, Services Computing and Intelligent Service Management
BPSC 2009 · ISM 2009 · YRW-MBP 2009
- P-148 Christian Erfurth, Gerald Eichler, Volkmar Schau (Eds.)
9th International Conference on Innovative Internet Community Systems
I²CS 2009
- P-149 Paul Müller, Bernhard Neumair, Gabi Dreö Rodosek (Hrsg.)
2. DFN-Forum
Kommunikationstechnologien
Beiträge der Fachtagung
- P-150 Jürgen Münch, Peter Liggesmeyer (Hrsg.)
Software Engineering
2009 - Workshopband
- P-151 Armin Heinzl, Peter Dadam, Stefan Kirn, Peter Lockemann (Eds.)
PRIMIUM
Process Innovation for Enterprise Software
- P-152 Jan Mendling, Stefanie Rinderle-Ma, Werner Esswein (Eds.)
Enterprise Modelling and Information Systems Architectures
Proceedings of the 3rd Int'l Workshop
EMISA 2009
- P-153 Andreas Schwill, Nicolas Apostolopoulos (Hrsg.)
Lernen im Digitalen Zeitalter
DeLFI 2009 – Die 7. E-Learning
Fachtagung Informatik
- P-154 Stefan Fischer, Erik Maehle, Rüdiger Reischuk (Hrsg.)
INFORMATIK 2009
Im Focus das Leben
- P-155 Arslan Brömmе, Christoph Busch, Detlef Hühnlein (Eds.)
BIOSIG 2009:
Biometrics and Electronic Signatures
Proceedings of the Special Interest Group on Biometrics and Electronic Signatures
- P-156 Bernhard Koerber (Hrsg.)
Zukunft braucht Herkunft
25 Jahre »INFOS – Informatik und Schule«
- P-157 Ivo Grosse, Steffen Neumann, Stefan Posch, Falk Schreiber, Peter Stadler (Eds.)
German Conference on Bioinformatics
2009
- P-158 W. Claupein, L. Theuvsen, A. Kämpf, M. Morgenstern (Hrsg.)
Precision Agriculture
Reloaded – Informationsgestützte Landwirtschaft
- P-159 Gregor Engels, Markus Luckey, Wilhelm Schäfer (Hrsg.)
Software Engineering 2010
- P-160 Gregor Engels, Markus Luckey, Alexander Pretschner, Ralf Reussner (Hrsg.)
Software Engineering 2010 –
Workshopband
(inkl. Doktorandensymposium)
- P-161 Gregor Engels, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.)
Modellierung 2010
- P-162 Maria A. Wimmer, Uwe Brinkhoff, Siegfried Kaiser, Dagmar Lück-Schneider, Erich Schweighofer, Andreas Wiebe (Hrsg.)
Vernetzte IT für einen effektiven Staat
Gemeinsame Fachtagung
Verwaltungsinformatik (FTVI) und
Fachtagung Rechtsinformatik (FTRI) 2010
- P-163 Markus Bick, Stefan Eulgem, Elgar Fleisch, J. Felix Hampe, Birgitta König-Ries, Franz Lehner, Key Pousttchi, Kai Rannenberg (Hrsg.)
Mobile und Ubiquitäre Informationssysteme
Technologien, Anwendungen und Dienste
zur Unterstützung von mobiler
Kollaboration
- P-164 Arslan Brömmе, Christoph Busch (Eds.)
BIOSIG 2010: Biometrics and Electronic Signatures
Proceedings of the Special Interest Group on Biometrics and Electronic Signatures

- P-165 Gerald Eichler, Peter Kropf, Ulrike Lechner, Phayung Meesad, Herwig Unger (Eds.)
10th International Conference on Innovative Internet Community Systems (I²CS) – Jubilee Edition 2010 –
- P-166 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
3. DFN-Forum Kommunikationstechnologien
Beiträge der Fachtagung
- P-167 Robert Krimmer, Rüdiger Grimm (Eds.)
4th International Conference on Electronic Voting 2010
co-organized by the Council of Europe, Gesellschaft für Informatik und E-Voting.CC
- P-168 Ira Diethelm, Christina Dörge, Claudia Hildebrandt, Carsten Schulte (Hrsg.)
Didaktik der Informatik
Möglichkeiten empirischer Forschungsmethoden und Perspektiven der Fachdidaktik
- P-169 Michael Kerres, Nadine Ojstersek, Ulrik Schroeder, Ulrich Hoppe (Hrsg.)
DeLFI 2010 - 8. Tagung der Fachgruppe E-Learning der Gesellschaft für Informatik e.V.
- P-170 Felix C. Freiling (Hrsg.)
Sicherheit 2010
Sicherheit, Schutz und Zuverlässigkeit
- P-171 Werner Esswein, Klaus Turowski, Martin Juhrisch (Hrsg.)
Modellierung betrieblicher Informationssysteme (MobIS 2010)
Modellgestütztes Management
- P-172 Stefan Klink, Agnes Koschmider, Marco Mevius, Andreas Oberweis (Hrsg.)
EMISA 2010
Einflussfaktoren auf die Entwicklung flexibler, integrierter Informationssysteme
Beiträge des Workshops der GI-Fachgruppe EMISA
(Entwicklungsmethoden für Informationssysteme und deren Anwendung)
- P-173 Dietmar Schomburg, Andreas Grote (Eds.)
German Conference on Bioinformatics 2010
- P-174 Arslan Brömmle, Torsten Eymann, Detlef Hühnlein, Heiko Roßnagel, Paul Schmücker (Hrsg.)
perspeGktive 2010
Workshop „Innovative und sichere Informationstechnologie für das Gesundheitswesen von morgen“
- P-175 Klaus-Peter Fährnrich, Bogdan Franczyk (Hrsg.)
INFORMATIK 2010
Service Science – Neue Perspektiven für die Informatik
Band 1
- P-176 Klaus-Peter Fährnrich, Bogdan Franczyk (Hrsg.)
INFORMATIK 2010
Service Science – Neue Perspektiven für die Informatik
Band 2
- P-177 Witold Abramowicz, Rainer Alt, Klaus-Peter Fährnrich, Bogdan Franczyk, Leszek A. Maciaszek (Eds.)
INFORMATIK 2010
Business Process and Service Science – Proceedings of ISSS and BPSC
- P-178 Wolfram Pietsch, Benedikt Krams (Hrsg.)
Vom Projekt zum Produkt
Fachtagung des GI-Fachausschusses Management der Anwendungsentwicklung und -wartung im Fachbereich Wirtschaftsinformatik (WI-MAW), Aachen, 2010
- P-179 Stefan Gruner, Bernhard Rumpe (Eds.)
FM+AM'2010
Second International Workshop on Formal Methods and Agile Methods
- P-180 Theo Härder, Wolfgang Lehner, Bernhard Mitschang, Harald Schöning, Holger Schwarz (Hrsg.)
Datenbanksysteme für Business, Technologie und Web (BTW) 14. Fachtagung des GI-Fachbereichs „Datenbanken und Informationssysteme“ (DBIS)
- P-181 Michael Clasen, Otto Schätzel, Brigitte Theuvsen (Hrsg.)
Qualität und Effizienz durch informationsgestützte Landwirtschaft, Fokus: Moderne Weinwirtschaft
- P-182 Ronald Maier (Hrsg.)
6th Conference on Professional Knowledge Management
From Knowledge to Action
- P-183 Ralf Reussner, Matthias Grund, Andreas Oberweis, Walter Tichy (Hrsg.)
Software Engineering 2011
Fachtagung des GI-Fachbereichs Softwaretechnik
- P-184 Ralf Reussner, Alexander Pretschner, Stefan Jähnichen (Hrsg.)
Software Engineering 2011
Workshopband
(inkl. Doktorandensymposium)

- P-185 Hagen Höpfner, Günther Specht,
Thomas Ritz, Christian Bunse (Hrsg.)
MMS 2011: Mobile und ubiquitäre
Informationssysteme Proceedings zur
6. Konferenz Mobile und Ubiquitäre
Informationssysteme (MMS 2011)
- P-186 Gerald Eichler, Axel Küpper,
Volkmar Schau, Hacène Fouchal,
Herwig Unger (Eds.)
11th International Conference on
Innovative Internet Community Systems
(I²CS)
- P-187 Paul Müller, Bernhard Neumair,
Gabi Dreö Rodosek (Hrsg.)
4. DFN-Forum Kommunikations-
technologien, Beiträge der Fachtagung
20. Juni bis 21. Juni 2011 Bonn
- P-189 Thomas, Marco (Hrsg.)
Informatik in Bildung und Beruf
INFOS 2011
14. GI-Fachtagung Informatik und Schule

The titles can be purchased at:

Köllen Druck + Verlag GmbH

Ernst-Robert-Curtius-Str. 14 · D-53117 Bonn

Fax: +49 (0)228/9898222

E-Mail: druckverlag@koellen.de